

**YOUR OWN PROGRAM**  
— the first time you try —



**CodeWriter**  
Corporation

For the Atari

# Adventure Writer™

A CodeWriter™ Program

Enjoy playing computer adventure games? Think about the thrill of creating them! Your own heroes and villains, magicians and monsters, castles and coffins... as many games as you can imagine. Create your adventures in plain English. AdventureWriter programs them on your own disks... to use, share, sell. And you don't need AdventureWriter to run your games.



# **AdventureWriter™**

for the Atari 800 and  
XL series computers with 48K



from CodeWriter Corporation in association with Gilsoft

written by Graeme Yeandle

© 1983 GILSOFT. Unauthorized copying, hiring or lending of  
the database Editor or of the Manual is strictly prohibited.



# **Acknowledgments**

## **Technical**

James Liu  
Steven Rutherford  
John Blume  
Mark Tanaka  
Christopher Rider

Alan Citterman  
Warren Shore  
Paul Helgeson  
Kevin Rutherford

**A special thanks to Graeme Yeandle and the master adventurers at Gilsoft**

## **Documentation**

Paul Helgeson  
Karen Hunter  
Alan Citterman  
John Blume

# Contents

---

## introduction

Chapter I	<b>Using This Manual</b>	
	Introduction Section	1
	Tutorial Section	1
	Reference Section	1
	The RESCUE	2
	Key to Symbols	2
	A Word About Software Protection	3
Chapter II	<b>Introducing AdventureWriter</b>	
	Definition: What is AdventureWriter?	5
	Description: What is an Adventure?	5
Chapter III	<b>Operation</b>	
	How AdventureWriter Works	7
	Standard Design Procedure	7
	Selling Your Created Adventures	9

---

## tutorial

Chapter IV	<b>Getting Started</b>	
	Adventure Objective	11
	Equipment and Materials Needed	11
	Preliminary Procedure	11
	Using AdventureWriter	12
Chapter V	<b>Phase I: Creating a Database</b>	
	Setting Up an Adventure	15
	Location Descriptions	15
	The Movement Table	21

	Object Descriptions	24
	Object Starting Locations	26
	Vocabulary Text Additions	28
	Decoding the Player's Commands	30
	The Vocabulary Action Table	31
 Chapter VI	<b>Phase II: Expanding the Database</b>	
	Adding Object Descriptions	39
	Changing Object Starting Locations	40
	Vocabulary Text Additions	42
	Adding Movement Table Entries	44
	Adding Message Text	46
	Adding Vocabulary Action Table Entries	48
	Status Table	53
 Chapter VII	<b>Phase III: Completing the Adventure</b>	
	Objective Descriptions	59
	Object Starting Locations	61
	The Movement Table	63
	The Vocabulary Action Table	66
	The Status Table	73

---

## reference

Chapter VIII	<b>Reference Section</b>	
	Detailed Description of the Database	81
	Detailed Description of the Editor	84
 Chapter IX	<b>Glossary</b>	
	AdventureWriter Glossary Part I	106
	AdventureWriter Main Menu Glossary	108
 Chapter X	<b>Index</b>	113





The AdventureWriter manual consists of 3 principal sections:

## **introductory section:**

This section gives a brief overview of what AdventureWriter is and what it does. Also included is a part called "Standard Procedure", a quick outline of the procedure you will use every time you design an adventure. You will find this outline to be particularly helpful if you are creating an elaborate adventure and forget just where you are within the creation process and what needs to be done next.

## **tutorial section:**

The tutorial provides a sample adventure for you to create by following step-by-step procedures. There are three phases, each expanding upon the development of previous ones to make your adventure more interesting. The procedures used in Phase II and III will be similar to those of Phase I. Be sure to read Chapter IV, "Getting Started", for information important to the success of your program! Symbols accompany each command to assist you. An explanation of these symbols is found in the next part of this chapter.

## **reference section:**

This section expands the program capabilities learned in the tutorial with a detailed description of AdventureWriter. We recommend that you read this section after you have worked through the tutorial and before you create your own adventure program. Included in this section are tables and explanations of the Interpreter, the Editor, and the Database. You will also find information on adventure sounds, adventure capacities and limitations, and diagnostic flags.

## THE RESCUE

We have also included a completed adventure entitled **Rescue** for you to experiment with and try to solve. This adventure was created from the AdventureWriter program. To play this adventure, select the 'p' option from the first menu. The object of the adventure is to rescue the princess and return her to the throne. Enjoy!

## KEY TO SYMBOLS



**keyboard** indicates that you are to type a word, command, or sentence involving multiple keystrokes rather than a single keystroke.



**single key** prompts a single entry: one letter, symbol, or number.



**function key** designates that a particular function key is to be struck: RETURN, SHIFT, CTRL, etc.



**paired symbols** indicate that a single keystroke is to be followed by striking a function key, usually the RETURN key.



**disk** signals a disk change. You will be required to remove the current disk from the drive and insert another one.



**screen** alerts you to what will appear on your computer screen following a particular action.

INIT HELLO

**underlined phrases** present a command or entry EXACTLY as you are to type it. These phrases must be typed in uppercase (capital letters) only.

**shaded box** displays a partial screen. Sometimes only one word or line changes on a screen after you have made an entry. Rather than presenting the entire screen again and again, we will present only that significant part of information that has changed.

## A WORD ABOUT SOFTWARE PROTECTION

CodeWriter Corporation has strong ideas about protecting software. We feel that both the software developer and the software customer have rights which must be protected. The developer must be protected from "unauthorized use" of his work. After all, if the marketplace does not reward the developer for his work, the work will not be produced, not be supported and not be improved.

But workable software protection cannot exclude the customer's rights. The paying customer makes all new software possible. Thus, the customer should be able to use the software freely and with confidence. A "back-up" copy of your AdventureWriter disk is available at a small cost (see the coupon included with your system). Also, a one year guarantee is part of your system cost. If your AdventureWriter disk ever fails to operate for any reason during this period, we'll replace it at no charge. Once your purchase is registered, you'll be notified of our help line for any questions you might have concerning AdventureWriter.

Have fun creating your own adventures with AdventureWriter. We have developed it to be the best!



## DEFINITION: What is AdventureWriter?

**AdventureWriter** is a CodeWriter program creation system which allows a person with no programming experience to create adventures in Assembly language.

## DESCRIPTION: What is an Adventure?

An **adventure** can be described as a computerized version of the game Dungeons and Dragons. In Dungeons and Dragons™ one person is nominated as the dungeon master, and he invents a dungeon for other players to explore to try to retrieve hidden treasures, usually protected by monsters of various shapes and sizes. Each player states his proposed actions to the dungeon master who decides on the outcome, sometimes with the help of dice to introduce a random element.

With an adventure, the computer takes the place of the dungeon master and the player or players explore a predefined dungeon. Most adventures will contain a **vocabulary** of words which the computer "understands", a variety of **locations** in which a player may wander around, and **objects** which have to be used in the correct way to enable the adventure to be solved. The computer will describe a situation to the player and invite him to decide on a course of **action**. The computer then tells the player the result of his action.

Everyone who plays an adventure has the problem of making the computer understand his **commands**. The computer will have only a limited vocabulary of perhaps a few hundred words and finding the right words can sometimes be a problem. For example, if you are playing Dungeons and Dragons™ and the dungeon master tells you "There is a lamp nearby", then if you decide to "Pick up the light", the dungeon master should know what you mean. If the same situation occurs when playing an adventure, the computer may understand "Get lamp" but may not know that "light" is **synonymous** with "lamp" or that "Pick up" means the same thing as "Get" on this

occasion. Even so, most players will very quickly get the knack of finding the correct words. It should be noted, however, that it is up to you, the adventure designer, to decide which words are to be included in the computer's vocabulary.

## HOW ADVENTUREWRITER WORKS

The AdventureWriter system is made up of three parts :

- A **Database** which contains all the information relevant to an adventure such as object descriptions, locations, messages, and a variety of tables.
- A database **Editor** which enables data to be inserted, amended or deleted from the database.
- An **Interpreter** (this is the "Dungeon master") that uses the data in the database to control the adventure.

## STANDARD DESIGN PROCEDURE

The procedure for designing an adventure may initially seem long and complex. Actually, the complexity is up to you and depends upon the kind of adventure you're developing. The procedures of inserting, altering, saving, etc. follow much the same pattern, and after the first several times become easy.

The principal challenge is to plot each ingredient and course of action methodically. Skipping a simple step could prove disastrous later in the program. For that reason we provide this outline as a quick reference to the steps in all their sequential tedium!

## I. Preliminary Procedures (on paper)

- A. Read the **entire** manual and work through the example adventure
- B. Draw a map of your adventure including location numbers and directions
- C. Make a list of the objects you want in your adventure
- D. Make a list of the messages needed in your adventure
- E. Make a list of all the words and their synonyms that you will use
- F. Decide which flags to use and their purposes
- G. Write down all the Vocabulary Action Table entries
  - Show conditions and actions
  - Make sure that you have entries with the same word value in the correct order
- H. Write down all the Status Table entries
  - Show conditions and actions
  - Plan the order of the entries. The Editor will arrange them in ascending order of word value, assigned when inserted in the Vocabulary Text Table
  - Make additions to the Vocabulary Text Table as needed

## II. Build Database (save Database regularly)

- A. Enter Location Descriptions
- B. Build Movement Table
- C. Enter Object Descriptions
- D. Enter Object Starting Locations
- E. Enter Messages
- F. Build Vocabulary (insert new words as needed)
- G. Build Vocabulary Action Table
  - Test Adventure (make changes as needed)
- H. Build Status Table
  - Test Adventure (make changes as needed)
- J. Expand Adventure (if desired)
  - Test Adventure (make changes as needed)
- I. Save Adventure
  - Test Adventure thoroughly
- K. Have Fun Adventuring !!!



## **SELLING YOUR CREATED ADVENTURES**

If you intend to sell your adventures, test them thoroughly with the help of as many people as possible.

In particular:

- A) Check the spelling of every word
- B) Check to see that the score does not exceed 100%
- C) Try to move in every direction from every location
- D) Try to GET, DROP, WEAR and REMOVE each object
- E) It should be possible to solve the adventure each time it is played, provided the correct commands are used.  
For example: poison gas which has a 1% chance of appearing and killing the player should be avoided unless you also provide a gas mask.

A great deal of time has been spent testing AdventureWriter and we believe that you should have no trouble creating hundreds of bug-free adventures. If you have any problems or questions, please contact us at CodeWriter and we'll be glad to give you a helping hand.

Thank you for your interest in the CodeWriter series of programs. Good luck with your Adventuring!

If you do intend to sell an adventure written with AdventureWriter, we require that you mention within the program that it was designed with CodeWriter Corporation's AdventureWriter Program.

We will be happy to consider marketing adventures you have written. If you are interested, call CodeWriter at (312) 470-0700 and ask about the Author's Program. Adventures must be accompanied by a completed submission form, a detailed map of all locations, and a list of all objects and vocabulary used.

**CodeWriter Corp.**  
7847 N. Caldwell  
Niles, IL 60648



Read this chapter thoroughly before beginning Phase I. It contains information needed to design and run an adventure program successfully.

The objective of our sample adventure "Thief" is to find and return the jewel to the living room in less than 21 turns.

## EQUIPMENT AND MATERIALS NEEDED

- Atari 800 or XL Series Computer with 48K
- Atari 810 or 1050 Disk Drive (or compatible)
- Monitor or TV
- Printer (optional)
- Diskettes:
  - AdventureWriter** Disk
  - Blank, formatted disk to contain your adventure program

## PRELIMINARY PROCEDURE

You will be creating a sample adventure program in this tutorial as a means of learning how to use the AdventureWriter Program. Before you begin the actual design process, you must have a blank, **formatted diskette** to contain the new program. The following instructions explain the procedure for formatting a disk.

- 1) **REMOVE** all cartridges from slots and **INSERT** Atari DOS 2.0 Master disk into drive
- 2) **POWER UP** the system. If the message "ready" appears, **TYPE** DOS and **PRESS** RETURN
- 3) **REMOVE** the DOS disk and **INSERT** disk to be formatted
- 4) **ENTER** 1 and **PRESS** RETURN then **ENTER** 1 and **PRESS** RETURN

5) **ENTER** Y and **PRESS** RETURN

Formatting will take about 40 seconds . . .

6) **ENTER** H and **PRESS** RETURN

7) **ENTER** I and **PRESS** RETURN

8) **ENTER** Y and **PRESS** RETURN

This concludes formatting.

## **USING ADVENTUREWRITER**

**REMOVE** all cartridges from the slots

**INSERT** the AdventureWriter disk into the drive

**POWER UP** computer

**ENTER** A from the menu to load AdventureWriter

You will see the credits, then:

**PRESS** any key to call up the Main Menu.

**REMOVE** the AdventureWriter disk and **INSERT** your application disk into the drive. No further disk swaps are required.

### **The Main Menu**

When AdventureWriter is loaded, you will be presented with the **Editor's Main Menu** which gives you a number of options. Some of these options (such as "Memory available") will perform a function and return you to the Main Menu, while others (such as "Location text") will give you a sub-menu. The "Exit AdventureWriter" option is an exception, as it executes a command which resets the computer.

## The Input Routine

AdventureWriter uses its own screen editor. When you type from the keyboard the characters are placed into a large storage area and the contents of that area are then printed on the screen. The reverse video key works in the usual way.

**CURSOR LEFT** and **CURSOR RIGHT** are used to move the cursor through the keyboard storage area. **CURSOR UP** and **CURSOR DOWN** are used to move the cursor within the keyboard storage area and will move it up to 40 characters or to the end of the line whichever comes first. **CLEAR** will clear the screen and the keyboard storage area.

To **insert** characters, simply position the cursor at the appropriate place and type in the characters to be inserted. It is not possible to type over characters that are already present. To **delete** characters, position the cursor to the right of the characters and press the back space key.

Whenever you press RETURN, the Editor checks your typing for **syntax**. If the Editor finds an invalid entry, it prints a "?" as the cursor immediately to the right of the error so that you may correct it. The only valid options on the Main Menu are **single capital letters** in the range **A** to **P** and **+**. Try typing in the number "3" and pressing RETURN, then delete the "3" and type in "ABC" followed by RETURN. In each case a syntax error will be detected.

## Memory Available

Let's now try one of the options on the Main Menu. if you clear the keyboard storage area (CLEAR), and type the letter **M** for Memory available and press RETURN, you will get a display which tells you how many **bytes** are unused in the database.

## Set Display Colors

When AdventureWriter is loaded, it has a blue screen with white text. The **colors** may be changed by selecting option **O** on the Main Menu. However, be careful not to make the text the same color as the screen.

## SAVE & LOAD DATABASE

**Save** and **load** are the options on the Main Menu which allow the database to be saved and reloaded. In each case, the computer will prompt you to "Type in name of file". When loading, the Atari will search for a file with the name you specified. A failure to load will be reported as an I/O ERROR.

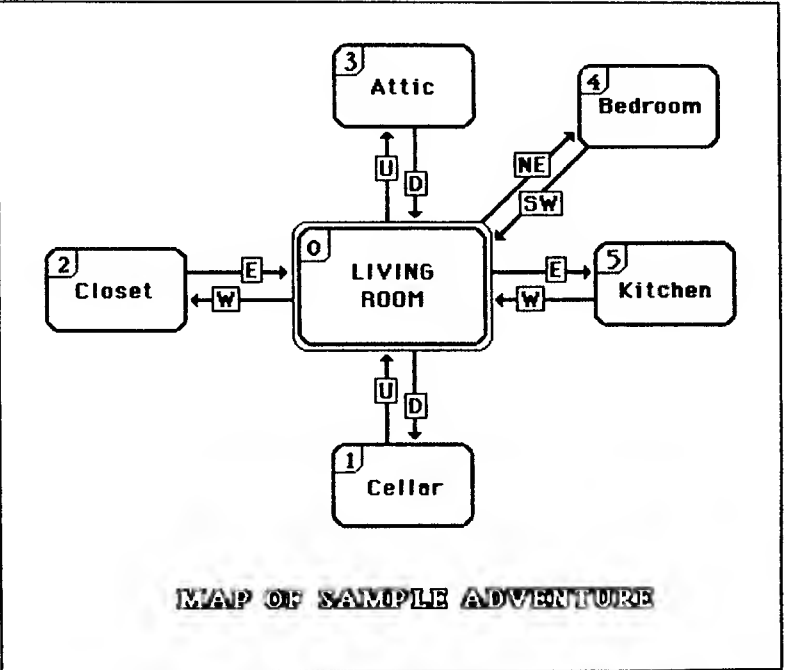
## **Phase I: Creating a Database**\_\_\_\_\_Chapter V

Phase I of the tutorial sets the stage for the entire adventure. The database you will create in this phase will be expanded in each successive phase. Consequently, you must methodically work through each step. If you wish to break part of the way through, please save the database so that you can continue later where you left off.

### **LOCATION DESCRIPTIONS**

The **location** setting is the core for playing any adventure. It is also the foundation for designing an adventure. Once established, objects, commands, and other intricacies are added. It is crucial that you **first** design your adventure locations on paper ! This map is an important reference for creating the rest of your adventure.

For the mini adventure you will create, we already provide you with a map:



The objective of the adventure is to find the jewel and return with it to the Living Room. The map shows all of the locations in the adventure and how they are inter-connected. Each location has been given a location number, shown in the corner.



ENTER C and PRESS RETURN

LOCATION DESCRIPTIONS

** Format **	** Description **
I	Insert desc
A <input type="text" value="loc"/>	Alter desc
P <input type="text" value="loc"/> optional	Print on Screen
L <input type="text" value="loc"/> optional	List to Printer
Z	Return to Main menu





ENTER P and PRESS RETURN



**Location 0**

I am in a prison cell. The walls are painted Black and Yellow and are perfectly smooth. The only way out is through a trapdoor in the ceiling which is 30 feet high. There is a **bright** light in one corner

Press any key to continue

You will see that a default description of location 0 already exists. We suggest that you highlight location names and directions to make your adventure easier to follow and a bit more exciting. Because a description for location 0 is already present, we must alter it to describe location 0 for our adventure (the living room).



ENTER any key

-



TYPE: A 0 and PRESS RETURN



**Location 0**

I am in a prison cell. The walls are painted Black and Yellow and are perfectly smooth. The only way out is through a trapdoor in the ceiling which is 30 feet high. There is a **bright** light in one corner ■



**PRESS** CLEAR (Depress Shift and '<' keys simultaneously)

You will type the following paragraph using the reverse video to highlight important words. **Bold** words represent words in inverse.

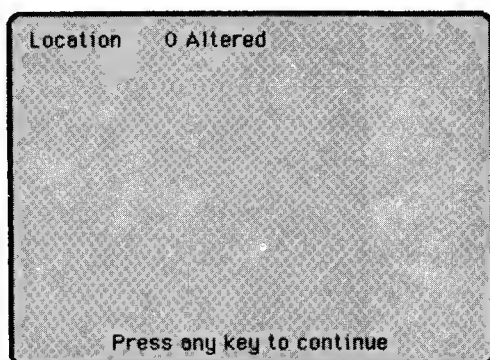
To achieve lower case letters, **PRESS** the CAPS/LOW key. To return to upper case letters, **PRESS** the SHIFT key and the CAPS/LOW key simultaneously.



**TYPE** I am in a **living room**. Stairs lead **Up** to the **attic** and **Down** to the **cellar**. A **closet** lies to the **West**. The **kitchen** is to the **East** and a **bedroom** is to the **NE**.



**PRESS** RETURN



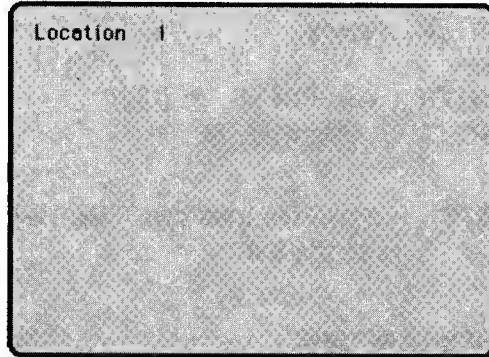


**ENTER** any key

Since there is no description for location 1 present, it must be inserted. After it is inserted it can be altered if you desire.



**ENTER** 1 and **PRESS** RETURN



**TYPE** I am in a damp **cellar**. Stairs lead **Up** to the **living room**.



**PRESS** RETURN



**ENTER** any key

Descriptions for the remaining locations must be inserted using the same format as location 1.



**ENTER** 1 and **PRESS** RETURN  
ON SCREEN: **location 2**



**TYPE** I am in the **closet**. The **living room** is directly to the **East**.



**PRESS** RETURN



**ENTER** any key



ENTER I and PRESS RETURN  
ON SCREEN: **location 3**



TYPE I am in a dusty **attic**. Stairs lead **Down**  
to the **living room**.



PRESS RETURN



ENTER any key



ENTER I and PRESS RETURN.  
ON SCREEN: **location 4**



TYPE I am in the **bedroom**. The **living room**  
is to the **SW**.



PRESS RETURN



ENTER any key



ENTER I and PRESS RETURN  
ON SCREEN: **location 5**



TYPE I am in the **kitchen**. The **living room**  
is directly to the **West**.



PRESS RETURN



ENTER any key

At this point you should have entered descriptions for  
locations 0-5. To see the location descriptions on screen:



ENTER P and PRESS RETURN

To send the location descriptions to your printer:



ENTER L and PRESS RETURN



To return to the main menu:  
ENTER Z and PRESS RETURN

### AdventureWriter Main Menu

- A ..... Vocabulary text
- B ..... Message text
- C ..... Location descriptions
- D ..... Movement table
- E ..... Object descriptions
- F ..... Object starting locations
- G ..... Vocabulary action table
- H ..... Status table
- I ..... Save a data base
- J ..... Load a data base
- K ..... Test this adventure
- L ..... Save this adventure
- M ..... Memory available
- N ..... \* of portable objects
- O ..... Set display colors
- P ..... AdventureWriter messages
- + ..... Exit AdventureWriter

Select an Option and Press **RETURN**

## THE MOVEMENT TABLE

The next step in creating our adventure is to interconnect the locations we have just described. This is done by means of the Movement Table.

**ENTER** D from the main menu and **PRESS** RETURN

### MOVEMENT TABLE

#### \*\* Format \*\*

A **loc \***  
P **loc \* optional**  
L **loc \* optional**

Z Return to Main menu

#### \*\* Description \*\*

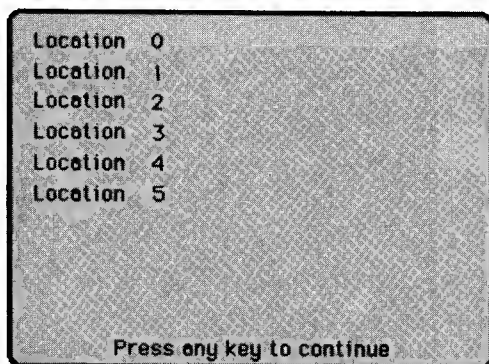
Alter an entry

Print on Screen

List to Printer

You will see that movements can be altered but not inserted. This is because the Editor automatically inserts null entries in the movement table for every entry made in the Location Description Table.

**ENTER** P and press **RETURN**



```
Location 0
Location 1
Location 2
Location 3
Location 4
Location 5

Press any key to continue
```

You can see that null entries for locations 0-5 are already present in the Movement Table.

**ENTER** any key

Refer to the map of our mini-adventure to see how the locations are interconnected. You will see the movements from location 0:

<b>Down</b>	goes to location 1 (cellar)
<b>West</b>	goes to location 2 (closet)
<b>Up</b>	goes to location 3 (attic)
<b>NE</b>	goes to location 4 (bedroom)
<b>East</b>	goes to location 5 (kitchen)

To enter these directions in the movement table:

**TYPE:** A 0 and **PRESS** **RETURN**

**TYPE:** D 1 W 2 U 3 NE 4 E 5

**PRESS** **RETURN**

**ON SCREEN:** location 0 altered



**ENTER** any key

Repeat this same procedure to alter the entries for locations 1-5.



**TYPE:** A 1 and **PRESS** RETURN

**TYPE:** U 0 and **PRESS** RETURN

**ON SCREEN:** location 1 altered

**ENTER** any key



**TYPE:** A 2 and **PRESS** RETURN

**TYPE:** E 0 and **PRESS** RETURN

**ON SCREEN:** location 2 altered

**ENTER** any key



**TYPE:** A 3 and **PRESS** RETURN

**TYPE:** D 0 and **PRESS** RETURN

**ON SCREEN:** location 3 altered

**ENTER** any key



**TYPE:** A 4 and **PRESS** RETURN

**TYPE:** SW 0 and **PRESS** RETURN

**ON SCREEN:** location 4 altered

**ENTER** any key



**TYPE:** A 5 and **PRESS** RETURN

**TYPE:** W 0 and **PRESS** RETURN

**ON SCREEN:** location 5 altered

**ENTER** any key



To see all the entries you altered in the movement table:



**ENTER** P and **PRESS** RETURN



Location 0	D	to	1
	W	to	2
	U	to	3
	NE	to	4
	E	to	5
Location 1	U	to	0
Location 2	E	to	0
Location 3	D	to	0
Location 4	SW	to	0
Location 5	W	to	0

Press any key to continue



ENTER any key



ENTER Z and PRESS RETURN to return to the Main menu

## OBJECT DESCRIPTIONS

The next step in creating our small adventure is to describe the objects we want to be present in our adventure. **Object descriptions** are entered in exactly the same way as location descriptions. The objects we want to include in the first phase of our adventure are:

An old hat  
A sparkling jewel  
A sharp knife  
A bottle of wine

To enter these descriptions, from the main menu:



ENTER E and PRESS RETURN



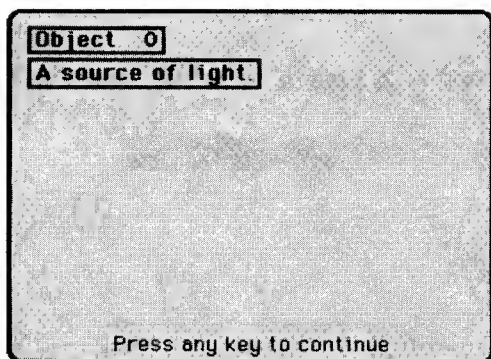
OBJECT DESCRIPTIONS	
** Format **	** Description **
I	Insert object desc
A <input type="text" value="obj"/>	Alter object desc
P <input type="text" value="obj * optional"/>	Print on Screen
L <input type="text" value="obj * optional"/>	List to Printer
Z	Return to Main menu

As with the location descriptions, there is a default object description for object 0. To see it:



ENTER P and PRESS RETURN





**Note:** The Interpreter always considers **object 0** to be a **source of light**. Object 0's presence in a dark location will enable the adventurer to see. We will use object 0 later in phase 3 so don't alter it yet.



**PRESS** any key

To enter a description for object 1:



**ENTER** 1 and **PRESS** RETURN

We suggest that you highlight the object descriptions (as you did with the location descriptions) to make it easier for the player to read and follow the adventure.



**TYPE** **An old hat** and **PRESS** RETURN

ON SCREEN: **Object 1 Inserted**



**PRESS** any key, **ENTER** 1 and **PRESS** RETURN



**TYPE** **A sparkling jewel** and **PRESS** RETURN

ON SCREEN: **Object 2 Inserted**



**PRESS** any key, **ENTER** 1 and **PRESS** RETURN



**TYPE** **A sharp knife** and **PRESS** RETURN

ON SCREEN: **Object 3 Inserted**



**PRESS** any key, **ENTER** 1 and **PRESS** RETURN



**TYPE**    **A bottle of wine**    and **PRESS** RETURN

ON SCREEN: **Object 4 Inserted.**

## OBJECT STARTING LOCATIONS

When you insert an object description the Editor automatically inserts a start location of 252 "not created" in the Object Starting Location Table. These starting locations must be altered for our adventure. We want:

The hat to start in the closet  
The jewel to start in the bedroom  
The knife to start in the kitchen  
The wine to start in the cellar

To do this from the Main menu:



**ENTER** E and **PRESS** RETURN

OBJECT STARTING LOCATIONS	
** Format **	** Description **
A <input type="text" value="obj"/> * <input type="text" value="loc"/> *	Alter starting loc of an object Special location #'s 252=Not created 253=Worn 254=Carried
P	Print on Screen
L	List to Printer
Z	Return to Main menu



**ENTER P and PRESS RETURN**

```
Object 0 not created
Object 1 not created
Object 2 not created
Object 3 not created
Object 4 not created
```

Press any key to continue



**PRESS any key**

To alter Object starting locations 1-4:

**TYPE A 1 2 and PRESS RETURN**

ON SCREEN: **Altered**



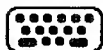
**PRESS any key, TYPE A 2 4 and PRESS RETURN**

ON SCREEN: **Altered**



**PRESS any key, TYPE A 3 5 and PRESS RETURN**

ON SCREEN: **Altered**



**PRESS any key, TYPE A 4 1 and PRESS RETURN**

ON SCREEN: **Altered**

**PRESS any key**

To see the alterations you made:



**ENTER P and PRESS RETURN**

```
Object 0 not created
Object 1 at location 2
Object 2 at location 4
Object 3 at location 5
Object 4 at location 1
```

Press any key to continue

Your Object Starting Locations should look exactly like the ones above. If they differ, alter them.

To return to the Main Menu:

**PRESS any key, ENTER Z and PRESS RETURN**



## VOCABULARY TEXT ADDITIONS

The **vocabulary** contains an entry for every word that the computer is to understand. Each entry consists of up to **four** letters followed by a number (or word value). Entries with the same word value are **synonyms**.

The vocabulary already contains over 30 words that will be used in most adventures. To see them:

ENTER A and PRESS RETURN

VOCABULARY TEXT	
** Format **	** Description **
I <u>word</u> <u>word</u> <u>word</u>	Insert a word
D <u>word</u>	Delete a word
S <u>word</u>	Show synonyms
P	Print on Screen
L	List to Printer
Z	Return to Main menu

ENTER P and PRESS RETURN

N	1
NORT	1
S	2
SOUT	2
E	3
EAST	3
W	4
WEST	4
NE	5
NW	6
SE	7
SW	8
U	9
UP	9
CLIM	9
ASCE	9
D	10
DOWN	10
DESC	10
GET	100
TAKE	100
DROP	101
REMO	102
More ...	

SE	7
SW	8
U	9
UP	9
CLIM	9
ASCE	9
D	10
DOWN	10
DESC	10
GET	100
TAKE	100
DROP	101
REMO	102
WEAR	103
I	104
INVE	104
R	105
REDE	105
QUIT	106
STOP	106
SAVE	107
LOAD	108

Press any key to continue

Notice that U, UP, CLIM, and ASCE all have a word value of 9. This shows that they are all synonyms and mean the same thing to the computer.

When you insert a word into the vocabulary, only the first four letters are inserted unless of course the word is less than four letters in length. The benefit is using less memory. To return to the Vocabulary Text Menu:

**PRESS** any key

You must make some additions to the vocabulary in order for the Editor to later understand the entries we have to make in the Vocabulary Action Table.

The words we want in our vocabulary are:

WINE, BOTTLE, HAT, JEWEL and KNIFE

**TYPE** WINE 21 and **PRESS RETURN**

ON SCREEN: WINE Inserted

**PRESS** any key, **TYPE** BOTTLE 21 and **PRESS RETURN**

ON SCREEN: BOTT Inserted

**PRESS** any key, **TYPE** HAT 22 and **PRESS RETURN**

ON SCREEN: HAT Inserted

**PRESS** any key, **TYPE** JEWEL 24 and **PRESS RETURN**

ON SCREEN: JEWEL Inserted

**PRESS** any key, **TYPE** KNIFE 25 and **PRESS RETURN**

ON SCREEN: KNIF Inserted

**PRESS** any key

To see the words you just added to the vocabulary:

**ENTER P** and **PRESS RETURN**

N	1
NORT	1
S	2
SOUT	2
E	3
EAST	3
W	4
WEST	4
NE	5
NW	6
SE	7
SW	8
U	9
UP	9
CLIM	9
ASCE	9
D	10
DOWN	10
DESC	10
WINE	21
BOTT	21
HAT	22
JEWEL	24

More ...

ASCE	9
D	10
DOWN	10
DESC	10
WINE	21
BOTT	21
HAT	22
JEWEL	24
KNIF	25
GET	100
TAKE	100
DROP	101
REMO	102
WEAR	103
I	104
INVE	104
R	105
REDE	105
QUIT	106
STOP	106
SAVE	107
LOAD	108

Press any key to continue

To return to the main menu:  
**ENTER** Z and **PRESS** RETURN

## DECODING THE PLAYER'S COMMANDS

Each time the player types in a command during an adventure the Interpreter has to decode it. Decoding is accomplished by searching the command for words which are in the Interpreter's vocabulary. The **word value** of the first word recognized is stored in a variable called **W1** and similarly the word value of the second word recognized is stored in **W2**.

This means that commands like TURN ON THE FLASHLIGHT can be reduced to ON FLAS provided that the words TURN and THE are not in the vocabulary and that ON and FLAS are included in the vocabulary. The interpreter scans the entire phrase, but will look for the first two recognizable words and try to act upon them. You can see that it is important to consider which words are excluded from the vocabulary as well as those which are included. The interpreter can only recognize two words per command and will only act upon the first two words it recognizes.

If no words are recognized, the Interpreter gives the reply "I don't understand...". If the Interpreter recognizes a word or words, but they neither cause movement (because of no entry in the Movement Table) nor cause an action to be performed (to be explained later), then the Interpreter gives one of the following replies.

"I can't"

when the value of W1 is  
greater than 12

"I can't go in that direction"

when the value of W1 is  
less than 13.

Therefore, the words in the vocabulary which relate to **directions** should have word values in the **range 1 to 12**.

## **THE VOCABULARY ACTION TABLE**

Both the Vocabulary Action Table and the Status Table are difficult concepts to understand, so this section of the manual is devoted to explaining how to use them.

The Vocabulary Action Table is created by you; it is used to tell the interpreter what to do when the player types in a command.

There are a few default entries already present in the table that will be used in most adventures. To see them from the main menu:

ENTER **G** and PRESS **RETURN**

VOCABULARY ACTION TABLE			
** Format **		** Description **	
A	<b>word1</b> <b>word2</b>		Alter entry
I	<b>word1</b> <b>word2</b>		Insert entry
P	<b>word1</b> <b>word2</b> opt		Print on Screen
L	<b>word1</b> <b>word2</b> opt		List to Printer
Z			Return to Main menu

We want to use the **P option** to print the table on the screen. Notice that you have the option of typing "P" followed by two words. If you enter only "P," the entire table will be listed. However, if your table is especially long, you may want a partial listing. Type two words in the places designated as "word 1" and "word 2." Only those words and the entries following will be displayed.

ENTER **P** and PRESS **RETURN**

GET	I	Conds	
		Acts	INVEN
I	*	Conds	
		Acts	INVEN
R	*	Conds	
		Acts	DESC
QUIT	*	Conds	
		Acts	QUIT
			END
SAVE	*	Conds	
		Acts	SAVE
LOAD	*	Conds	
		Acts	LOAD

Press any key to continue



Notice the entries that are already present. These are built-in commands that you will want in most adventures. Their functions will be explained a little later.

Look at the two left columns on the screen. The first column holds one word and the second column holds either a word or an asterisk (\*). **WORD/WORD** combinations prompt the Interpreter to look for two words. **WORD/\*** combinations prompt the Interpreter to look for a one-word command and the player need enter only one word for the action to take place. Any words following it will be ignored.

### **Action INVEN**

This action will print "I have with me: - " and give a list or **inventory** of what the player has with him.

### **Action DESC**

This action will clear the screen and attempt to **describe** the current location. If it is dark the Interpreter will print "Everything is dark. I can't see."

### **Actions SAVE and LOAD**

These actions will copy and restore a game position. For example, if a player is in the middle of an adventure but doesn't have time to continue playing, he can **save** his current position within the game and **load** it later when he has more time to play the game.

### **Actions QUIT, TURNS and END**

The **QUIT** action will print "Are you sure you want to quit now?".

The action **TURNS** will print "You have taken x turn(s).".

The action **END** is very important, it prints "END OF GAME Do you want to try again?". If the player replies 'N' he is returned to the Editor. The only way to return to the Editor, after testing an adventure is to include the action **END** in the Vocabulary Action Table.

The first line in the Vocabulary Action Table should read:

**GET   I       CONDS**

Translation: Get or Take Inventory of the player's  
Objects according to Conditions Set (if any)

If the first line does not appear as shown here, you have added some lines of your own into the table.

The first two words represent a command that the player types in. These words are present in the list of Vocabulary Text (option A from the Main Menu). You could enter "TAKE I" rather than "GET I," because the Editor knows which words are synonyms.

If the player were to type the command "TAKE INVE," the Interpreter would check each entry in the Vocabulary Action Table until it found comparable words, "GET I." In this case, these words are the first ones in the list.

After the Interpreter finds the entry, it checks for conditions before attempting to act upon the words typed in by the player. Since in this example there is nothing following "CONDS," the Interpreter knows there are no conditions and it will proceed to act upon the entered words.

The next line should read "ACTS INVEN." The Interpreter considers anything following "ACTS" to be an action or actions. The Interpreter knows what to do when it encounters "INVEN" and it prints a list of objects that the player has with him.

Let's stop and take a look at the entire entry. When the player types "TAKE INVE" (or any equivalent), the Interpreter checks for any conditions, finds none and then takes the action INVEN.

The next entry in the table should read:

**I   \*       CONDS**

Translation: Take Inventory of any and all of the player's  
Objects according to Conditions Set (if any)

This entry causes the Interpreter to perform the inventory action. Functionally, it performs the same way for this action entry as for the previous entry GET 1 CONDS.

The "I" is a synonym of "INVE." The "\*" is like a wild card; it stands for any word or even no word at all. If the player were to type "I JUNK," "I," "INVENTORY" or "INVE CARRIED" it would all mean the same thing to the Interpreter.

There are no conditions for "I \*," and the only action is "INVE." Combining this and the first entry in the table, you can see that typing "INVE," "TAKE INVE," "I," or "I JUNK" will all result in the same thing: a list of what the player is carrying.

Look over the other entries in the table to see what they do. A table of conditions and actions is included in the Reference Section to assist in your understanding of this table.

Now it is time for you to make some additions to the Vocabulary Action Table for our adventure. We want the player to be able to "GET" and "DROP" all the objects in our adventure.



TYPE I GET WINE and PRESS RETURN



TYPE GET 4 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key, TYPE I GET HAT and PRESS RETURN



TYPE GET 1 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key, TYPE I GET JEWEL and PRESS RETURN



TYPE GET 2 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key, TYPE I GET KNIFE and PRESS RETURN



TYPE GET 3 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key, TYPE I DROP WINE and PRESS RETURN

TYPE DROP 4 OK and PRESS RETURN  
ON SCREEN: Inserted

PRESS any key, TYPE I DROP HAT and PRESS RETURN



TYPE DROP 1 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key, TYPE I DROP JEWEL and PRESS RETURN



TYPE DROP 2 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key, TYPE I DROP KNIFE and PRESS RETURN



TYPE DROP 3 OK and PRESS RETURN  
ON SCREEN: Inserted



PRESS any key

We are finished with our entries in the Vocabulary Action Table for Phase 1 of our adventure. To see the entries:



ENTER P and PRESS RETURN



GET WINE	Conds	GET 4
	Acts	OK
GET HAT	Conds	GET 1
	Acts	OK
GET JEWE	Conds	GET 2
	Acts	OK
GET KNIF	Conds	GET 3
	Acts	OK
GET I	Conds	INVEN
	Acts	
DROP WINE	Conds	DROP 4
	Acts	OK
DROP HAT	Conds	DROP 1
	Acts	OK
DROP JEWE	Conds	DROP 2
	Acts	
More ...		



DROP	HAT	Conds	OK	
		Acts	DROP	1
			OK	
DROP	JEWE	Conds		
		Acts	DROP	2
			OK	
DROP	KNIF	Conds		
		Acts	DROP	3
			OK	
I	*	Conds		
		Acts	INVENT	
R	*	Conds		
		Acts	DESC	
QUIT	*	Conds		
		Acts	QUIT	
			URNS	
			END	
SAVE	*	Conds		
		Acts	SAVE	
LOAD	*	Conds		
		Acts	LOAD	
Press any key to continue				

If your table does not look like the one shown above make the appropriate changes.

To return to the Main Menu:



**ENTER** Z and **PRESS** RETURN

de Nume  
Gloria

## Phase II:

## Expanding a Database \_\_\_\_\_ Chapter VI

### ADDING OBJECT DESCRIPTIONS

Phase II expands on the adventure developed in Phase I. In this section you will add a few objects to your adventure. The objects are :

- An open safe
- A big brass key
- A closed safe

Notice the safe is considered as two objects, but both are not allowed to be present at the same time. You will start with a closed safe; when the player opens the safe the closed safe is "destroyed" and an open safe is "created." Don't worry if you don't understand what we're saying right now; It will become clear later.

To Insert the new objects in the Object description table from the Main menu:



ENTER E and PRESS RETURN

OBJECT DESCRIPTIONS	
** Format **	** Description **
I	Insert object desc
A <u>obj *</u>	Alter object desc
P <u>obj * optional</u>	Print on Screen
L <u>obj * optional</u>	List to Printer
Z	Return to Main menu



ENTER I and PRESS RETURN  
ON SCREEN: Object 5



TYPE An open safe and PRESS RETURN  
ON SCREEN: Inserted



**PRESS** any key, **ENTER** 1 and **PRESS** RETURN  
ON SCREEN: **Object 6**



**TYPE** A big brass key and **PRESS** RETURN  
ON SCREEN: **Inserted**



**PRESS** any key, **ENTER** 1 and **PRESS** RETURN  
ON SCREEN: **Object 7**



**TYPE** A closed safe and **PRESS** RETURN  
ON SCREEN: **Inserted**



**PRESS** any key

To return to the Main Menu:



**ENTER** Z and **PRESS** RETURN

## **CHANGING OBJECT STARTING LOCATIONS**

Since you've added objects to your adventure, you must make some changes in the Object Starting Locations Table. Remember that the Editor automatically assigns a location number of 252 to each object described. The number 252 means that the starting location for an object has not yet been created or defined.

We want **object 5** (An open safe) to remain "not created" so do not alter it.

**Object 6** (A big brass key) will start the adventure in the attic so you must alter it.

**Object 7** (A closed safe) will start in the bedroom and must also be altered.



**Object 2** (A sparkling jewel) will start in the closed safe and will not be created until the safe is opened.

To make these changes from the Main menu:

**ENTER** E and **PRESS** RETURN

OBJECT STARTING LOCATIONS	
** Format **	** Description **
A <input type="text" value="obj"/> <input type="text" value="loc"/>	Alter starting loc of an object. Special location #'s 252=Not created 253=Worn 254=Carried
P	Print on Screen
L	List to Printer
Z	Return to Main menu

**TYPE** A 2 252 and **PRESS** RETURN

ON SCREEN: **Altered**

**PRESS** any key

**TYPE** A 6 3 and **PRESS** RETURN

ON SCREEN: **Altered**

**PRESS** any key

**TYPE** A 7 4 and **PRESS** RETURN

ON SCREEN: **Altered**

**PRESS** any key

To see the Altered Object Starting Locations:

**PRESS** P and RETURN

```
Object 0 not created
Object 1 at location 2
Object 2 not created
Object 3 at location 5
Object 4 at location 1
Object 5 not created
Object 6 at location 3
Object 7 at location 4
```

Press any key to continue

If your entries don't look like the ones above make the appropriate changes.

**Note:** These locations mark the starting point for the objects in the adventure. Their locations will change throughout the adventure as the player carries them around or destroys them.

To return to the Main Menu:

**ENTER** Z and **PRESS** RETURN

## VOCABULARY TEXT ADDITIONS

Since you are going to make some improvements to your adventure in this phase, you must also expand the Interpreter's vocabulary so it can understand the entries in other tables.

You will add these words:

**LIVI** (living room)  
**CELL** (cellar)  
**ATTI** (attic)  
**BEDR** (bedroom)  
**KITC** (kitchen)  
**SAFE**  
**KEY**  
**FINI** (finish)  
**OPEN**  
**UNLO** (unlock)  
**DRIN** (drink)

These entries must be present in the vocabulary in order to use them in the **movement**, **status**, and **vocabulary action tables**. To make these additions from the Main menu:



ENTER A and PRESS RETURN



VOCABULARY TEXT	
** Format **	** Description **
I <u>word</u> <u>word</u> *	Insert a word
D <u>word</u>	Delete a word
S <u>word</u>	Show synonyms
P	Print on Screen
L	List to Printer
Z	Return to Main menu



TYPE I LIVI 30 and PRESS RETURN

ON SCREEN: LIVI Inserted

PRESS any key



TYPE I CELL 31 and PRESS RETURN

ON SCREEN: CELL Inserted

PRESS any key



TYPE I CLOS 32 and PRESS RETURN

ON SCREEN: CLOS Inserted

PRESS any key



TYPE I ATTI 33 and PRESS RETURN

ON SCREEN: ATTI Inserted

PRESS any key



TYPE I BEDR 34 and PRESS RETURN

ON SCREEN: BEDR Inserted

PRESS any key



TYPE I KITC 35 and PRESS RETURN

ON SCREEN: KITC Inserted

PRESS any key

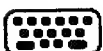




TYPE I SAFE 50 and PRESS RETURN  
 ON SCREEN: SAFE Inserted  
 PRESS any key



TYPE I KEY 51 and PRESS RETURN  
 ON SCREEN: KEY Inserted  
 PRESS any key



TYPE I FINI 55 and PRESS RETURN  
 ON SCREEN: FINI Inserted  
 PRESS any key



TYPE I OPEN 60 and PRESS RETURN  
 ON SCREEN: OPEN Inserted  
 PRESS any key



TYPE I UNLO 60 and PRESS RETURN  
 ON SCREEN: UNLO Inserted  
 PRESS any key



TYPE I DRIN 65 and PRESS RETURN  
 ON SCREEN: DRIN Inserted  
 PRESS any key



If you want to see the words you just inserted:  
 ENTER P and PRESS RETURN

## ADDING MOVEMENT TABLE ENTRIES

You will make some additions to the **movement table** to improve your adventure. You will enable the player to move from one location to any other adjacent location by simply typing in the name of that location. If you look at the map you'll see that all locations are adjacent to location 0. To make these additions from the main menu:



ENTER D and PRESS RETURN

MOVEMENT TABLE		
** Format **		** Description **
A	<u>loc</u> *	Alter an entry
P	<u>loc</u> * <u>optional</u>	Print on Screen
L	<u>loc</u> * <u>optional</u>	List to Printer
Z		Return to Main menu



TYPE A 0 and PRESS RETURN

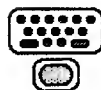
ON SCREEN: D 1 W 2 U 3 NE 4 E 5

Continuing on the same line:

TYPE ATT 1 3 CLOS 2 CELL 1 KITC 5 BEDR 4

PRESS RETURN

ON SCREEN: Altered



PRESS any key

TYPE A 1 and PRESS RETURN

ON SCREEN: U 0



On the same line TYPE LIVI 0 and PRESS RETURN

ON SCREEN: Altered



PRESS any key

TYPE A 2 and PRESS RETURN

ON SCREEN: E 0



On the same line TYPE LIVI 0 and PRESS RETURN

ON SCREEN: Altered



PRESS any key

TYPE A 3 and PRESS RETURN

ON SCREEN: D 0



On the same line TYPE LIVI 0 and PRESS RETURN

ON SCREEN: Altered





PRESS any key

TYPE A 4 and PRESS RETURN

ON SCREEN: SW 0



On the same line TYPE LIVI 0 and PRESS RETURN

ON SCREEN: Altered



PRESS any key

TYPE A 5 and PRESS RETURN

ON SCREEN: W 0



On the same line TYPE LIVI 0 and PRESS RETURN

ON SCREEN: Altered

To return to the Main Menu:



ENTER Z and PRESS RETURN

## ADDING MESSAGE TEXT

To make our adventure more interactive you will add some messages to be displayed in response to the player's actions. The **message text** uses the same format as the location descriptions, except that there is a **message number** instead of location number. A default message 0 is already present; to see it from the Main menu:



ENTER B and PRESS RETURN

MESSAGE TEXT	
** Format **	** Description **
I	Insert a msg
A <u>msg *</u>	Alter a msg
P <u>msg * optional</u>	Print on Screen
L <u>msg * optional</u>	List to Printer
Z	Return to Main menu



ENTER P and PRESS RETURN  
ON SCREEN: 'I see nothing special'  
To alter the default message 0:



PRESS any key  
TYPE A 0 and PRESS RETURN



To clear the entry PRESS SHIFT and CLEAR  
TYPE You win!!! Congratulations!!! and PRESS RETURN  
PRESS any key

Since there is no text present for messages 1 and 2, it must be inserted. To do this:



TYPE 1 and PRESS RETURN  
ON SCREEN: Message 1



TYPE Yum. An excellent wine. and PRESS RETURN  
PRESS any key

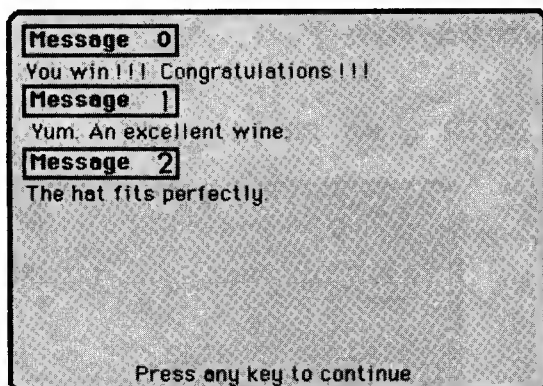


TYPE 1 and PRESS RETURN  
ON SCREEN: Message 2



TYPE The hat fits perfectly. and PRESS RETURN  
PRESS any key

To see the messages you entered:  
ENTER P and PRESS RETURN



## ADDING VOCABULARY ACTION TABLE ENTRIES

In the first phase of your adventure you didn't specify any conditions for the entries inserted in the **vocabulary action table**. You enabled the player to "GET," "DROP" objects. With these commands, the player carries objects from one location to another. The Interpreter knows what objects are present at each location and what objects the player is carrying or wearing. As a result, you don't have to check for these conditions in most entries.

However, the entries will get more complicated. For instance: we don't want the player to be able to open the safe unless he is carrying the key (object 6) and a closed safe (object 7) is present at the current location. (You can't open a safe that is already open). So we'll specify these as the conditions.

When these conditions are met, we want to destroy the closed safe, create the open safe and create the jewel. We'll also give him a score of 50% for opening the safe and describe the location for him again (since the objects present will have changed).

To Insert these entries into the Vocabulary Action Table from the main menu:

ENTER G and PRESS RETURN

VOCABULARY ACTION TABLE		
** Format **		** Description **
A	<u>word1</u> <u>word2</u>	Alter entry
I	<u>word1</u> <u>word2</u>	Insert entry
P	<u>word1</u> <u>word2</u> opt	Print on Screen
L	<u>word1</u> <u>word2</u> opt	List to Printer
Z	Return to Main menu	





TYPE I OPEN SAFE and PRESS RETURN



TYPE PRESENT 7 CARRIED 6 DESTROY 7 CREATE 5 CREATE 2  
LET 30 50 DESC



PRESS RETURN

ON SCREEN: **I OPEN SAFE Inserted**



PRESS any key

Before we go further let's examine this entry:



TYPE P OPEN SAFE and PRESS RETURN



OPEN SAFE	Conds	PRESENT	7
	Conds	CARRIED	6
	Acts	DESTROY	7
		CREATE	5
		CREATE	2
		LET	30 50
		DESC	
GET WINE	Conds	GET	4
	Acts	OK	
GET HAT	Conds	GET	1
	Acts	OK	
GET JEWE	Conds	GET	2
	Acts	OK	
GET KNIF	Conds	GET	3
	Acts	OK	
GET I	Conds	INVEN	
DROP WINE	Acts		
	Conds		
More .			

Your entry should match the one shown above. If not, make the appropriate changes.

Notice that we specified two conditions:

**PRESENT 7**  
**CARRIED 6**

Satisfied IF there is a closed  
safe and IF he is carrying the key

We specified five **actions**:

<b>DESTROY</b>	<b>7</b>	<b>THEN</b> destroy the closed safe
<b>CREATE</b>	<b>5</b>	create the open safe
<b>CREATE</b>	<b>2</b>	create the jewel.
<b>LET</b>	<b>30 50</b>	give them a score of 50 (flag 30 holds the player's score)
<b>DESC</b>		describe the location.

**NOTE:** Only if all the conditions for an entry are met will the actions for that entry be performed.

We have a few more entries to insert so:

**PRESS** any key

In our adventure the player has to carry (GET) the wine before he can drink it. We also print a message when he does drink it.

**TYPE** I DRINK WINE and **PRESS** RETURN

**TYPE** CARRIED 4 MESSAGE I OK and **PRESS** RETURN

ON SCREEN: I DRINK WINE Inserted

**PRESS** any key

We need an entry for the player to be able to "GET" the key:

**TYPE** I GET KEY and **PRESS** RETURN

**TYPE** GET 6 OK and **PRESS** RETURN

ON SCREEN: GET KEY Inserted

**PRESS** any key

**TYPE** I DROP KEY and **PRESS** RETURN

**TYPE** DROP 6 OK and **PRESS** RETURN

ON SCREEN: I DROP KEY Inserted

**PRESS** any key



TYPE I WEAR HAT and PRESS RETURN



TYPE WEAR 1 MESSAGE 2 OK and PRESS RETURN  
ON SCREEN: I WEAR HAT Inserted



PRESS any key



TYPE I REMOVE HAT and PRESS RETURN



TYPE REMOVE 1 OK and PRESS RETURN  
ON SCREEN: I REMOVE HAT Inserted



PRESS any key

To see the entries you made in the table:



ENTER P and PRESS RETURN



OPEN SAFE	Conds	PRESENT	7	
	Conds	CARRIED	6	
	Acts	DESTROY	7	
		CREATE	5	
		CREATE	2	
		LET	30	50
		DESC		
DRIN WINE	Conds	CARRIED	4	
	Acts	MESSAGE	1	
		OK		
GET WINE	Conds	GET	4	
	Acts	OK		
GET HAT	Conds	GET	1	
	Acts	OK		
GET JEWEL	Conds	GET	2	
	Acts	OK		
GET KNIF	Conds	GET	3	
	Acts	OK		
More ...				

GET KEY	Conds Acts	OK	
GET I	Conds Acts	GET OK	6
DROP WINE	Conds Acts	INVEN	
DROP HAT	Conds Acts	DROP OK	4
DROP JEWE	Conds Acts	DROP OK	1
DROP KNIF	Conds Acts	DROP OK	2
DROP KEY	Conds Acts	DROP OK	3
REMO HAT	Conds Acts	DROP OK	6
		REMOVE	1
More			

DROP KEY	Conds Acts	DROP OK	6
REMO HAT	Conds Acts	REMOVE OK	
WEAR HAT	Conds Acts	WEAR MESSAGE 2 OK	1
I *	Conds Acts	INVEN	
R *	Conds Acts	DESC	
QUIT *	Conds Acts	QUIT TURNS END	
SAVE *	Conds Acts	SAVE	
LOAD *	Conds Acts	LOAD	
Press any key to continue			

If your entries don't match the ones shown above, make the appropriate changes.

## STATUS TABLE

The **Status Table** has exactly the same format as the Vocabulary Action Table. Each entry consists of **two words** (both present in the vocabulary). As before, the second word can be an asterisk (\*). The words are followed by conditions and actions. However, the Interpreter uses this table in a different way.

The Status Table is scanned between each turn regardless of what the player types in. Every entry in the table is examined and if the conditions for an entry are satisfied, then the actions that follow are performed. A typical entry in the table would determine if a player has won the game.

In our adventure a player wins when he returns to the living room (location 0) with the jewel (object 2). When these conditions are satisfied, we want to increase the player's score, make a winning sound, print a winning message, display the score, display the number of turns taken, and end. This may sound complicated but it's really not.

We specify two conditions

AT	0	Satisfied IF he's in the
PRESENT	2	living room and IF the
		jewel is present

We perform these **actions**:

**PLUS** 30 50 add 50 to the score

**SOUND** 0 90 10 10

**PAUSE** 10

**SOUND** 0 70 10 10

**PAUSE** 10

**SOUND** 0 90 10 10

**PAUSE** 10

**SOUND** 0 60 10 10

**PAUSE** 15

**SOUND** 0 0 0 0

**MESSAGE** 0 print "you win !!!"

**SCORE** display score (flag 30)

**TURNS** display number of turns

**END** print "Do you want to play again?"

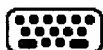
To Insert this entry, from the Main menu:

**ENTER** H and **PRESS** RETURN

STATUS			
** Format **		** Description **	
A	<b>word1</b> <b>word2</b>	Alter entry	
I	<b>word1</b> <b>word2</b>	Insert entry	
P	<b>word1</b> <b>word2</b> opt	Print on Screen	
L	<b>word1</b> <b>word2</b> opt	List to Printer	
Z	Return to Main menu		



TYPE I FINISH \* and PRESS RETURN



TYPE: AT 0 PRESENT 2 PLUS 30 50 SOUND 0 90 10 10  
 PAUSE 10 SOUND 0 70 10 10 PAUSE 10 SOUND  
 0 70 10 10 PAUSE 10 SOUND 0 90 10 10 PAUSE  
 10 SOUND 0 60 10 10 PAUSE 15 SOUND 0 0 0 0  
 MESSAGE 0 SCORE TURNS END



PRESS RETURN and PRESS any key

To see the entry:



ENTER P and PRESS RETURN



FINI *	Conds	AT	
		PRESENT	0
		PLUS	2
		PLUS	30 50
		SOUND	0 90 10 10
		PAUSE	10
		SOUND	0 70 10 10
		PAUSE	10
		SOUND	0 90 10 10
		PAUSE	10
		SOUND	0 60 10 10
		PAUSE	15
		SOUND	0 0 0 0
		MESSAGE	0
		SCORE	
		TURNS	
		END	

Press any key to continue

Your entry should match the one shown above. If not, make the appropriate changes.

**Note:** The words "FINISH \*" are used primarily as a reminder to yourself of what that entry does. We could have used the words "WIN \*" instead. The Interpreter does not look at these words, only the conditions that follow them. When the conditions are met, the actions that ensue will be performed.

**Note:** When you insert entries into the Status Table, the Editor looks at the **word values** of W1 and W2. (Every word in the vocabulary has a corresponding value.) It then enters them into the table in ascending numerical order. Consequently, you can position an entry at a particular place in the table by using words with the proper values.

To return to the Main menu:



**ENTER** Z and **PRESS** RETURN

### Save the Database

At this point you have entered all the database additions for phase II of our adventure. **save** these changes as a separate database. From the Main menu:

**ENTER** I and **PRESS** RETURN

ON SCREEN: **Type in name of file**

**TYPE:** PHASEII and **PRESS** RETURN

When the database has been saved correctly, you will be prompted to press any key to continue. If an error message is displayed, resave the database until it is saved correctly.

### Test the Adventure

Now it is time to **test** PHASE II of the adventure. From the Main menu:



**ENTER** K and **PRESS** RETURN

ON SCREEN: **Do you want to see diagnostic flags?**

The flags are used in the more advanced stages of testing your adventures. You will not need to see the flags for this phase of the adventure.

**PRESS** RETURN



The screen displays a description of location 0. Try moving in all directions from every room. Make sure that you are at the correct location after each move. You shouldn't be able to open the safe without the key and you shouldn't be able to get the jewel without opening the safe. When you are in the living room type in the following commands:

<b>INVENTORY</b>	will reply	I have with me:- Nothing at all.
<b>GET HAT</b>	will reply	It's not here.
<b>CLOSET</b>	will display	Description of the closet
<b>GET HAT</b>	will reply	OK
<b>WEAR HAT</b>	will reply	The hat fits perfectly.
<b>INVENTORY</b>	will reply	I have with me:- An old hat.(worn)

Since you added the entry to allow the player win, you now have two ways to exit from testing the adventure. If you get the jewel and returning with it to the living room you will win and will have the option to exit or to try again. Of course you can enter the command QUIT or STOP as you did when testing PHASE I.



## Phase III:

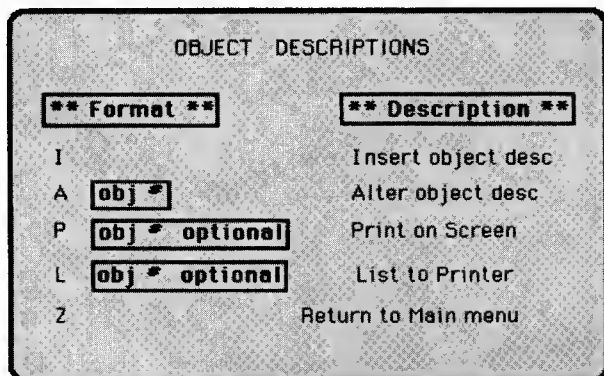
## Completing the Adventure \_\_\_\_\_ Chapter VII

### OBJECT DESCRIPTIONS

In this phase of the adventure we're going to make the attic dark so it's a little harder for the player to find the key. If you remember, earlier we said that the Interpreter always considers object 0 to be a source of light, and its presence at a dark location will enable the player to see.

We're going to need object 0 in this phase, so we'll describe it as "A flashlight (on)." To do this from the Main menu:

ENTER E and PRESS RETURN



** Format **	** Description **
I	Insert object desc
A <u>obj</u>	Alter object desc
P <u>obj</u> * optional	Print on Screen
L <u>obj</u> * optional	List to Printer
Z	Return to Main menu

Since an entry for object 0 is already present we will have to alter it:

TYPE A 0 and PRESS RETURN

ON SCREEN: A source of light. (the existing entry)

PRESS SHIFT and CLEAR to clear the entry

TYPE A flashlight. (on) and PRESS RETURN

ON SCREEN: Object 0 Altered

PRESS any key

We really don't need the knife in this adventure, so let's alter it to be "A flashlight. (off)". To do this:



TYPE A 3 and PRESS RETURN  
ON SCREEN: A sharp knife. (the existing entry)



PRESS SHIFT and CLEAR to clear the entry



TYPE A flashlight. (off) and PRESS RETURN  
ON SCREEN: Object 3 Altered

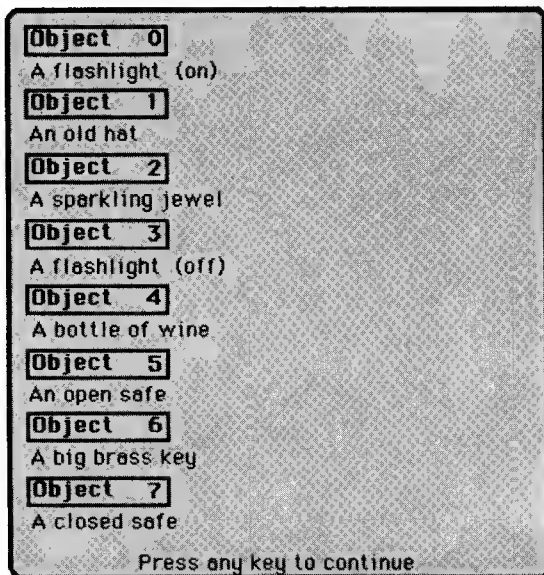


PRESS any key

To see the entries:



ENTER P and PRESS RETURN



If your entries don't match the ones shown above, make the appropriate changes.

## OBJECT STARTING LOCATIONS

In this part of your adventure you should not have to make any changes to the starting locations. You'll need the flashlight that is on (object 0) to start the adventure as "not created." The flashlight that is off (object 3) should start the adventure in the kitchen (location 5). Check the entries in your table to see that these objects start in the correct locations.

When you "turn on" the flashlight in the vocabulary action table, you'll swap (change the positions of) objects 0 and 3.

To return to the Main Menu:



PRESS any key, ENTER Z and PRESS RETURN

## THE VOCABULARY TEXT

It will be necessary, however, to add more words to the vocabulary. These entries must be made before any of these words can be used in any of the tables. The words we want to add to the vocabulary are:

FLASHLIGHT  
LIGHT  
ON  
OFF  
LOSE  
THIRST

Since you don't have a knife in your adventure any more, you don't need it in the vocabulary, and you'll delete it:



From the Main Menu:

**ENTER** Δ and **PRESS** RETURN

VOCABULARY TEXT	
** Format **	** Description **
I <u>word</u> <u>word</u> *	Insert a word
D <u>word</u>	Delete a word
S <u>word</u>	Show synonyms
P	Print on Screen
L	List to Printer
Z	Return to Main menu



**TYPE** D KNIFE and **PRESS** RETURN

ON SCREEN: KNIF DELETED

**PRESS** any key

**NOTE:** At the same time that "KNIF" was deleted from the vocabulary, the Editor automatically deleted the "GET KNIF" and "DROP KNIF" entries from the vocabulary action table. If there had been any "KNIF" entries in the status or movement tables they would have been deleted as well.

To insert the preceding words in the vocabulary:



**TYPE** I FLASH 25 and **PRESS** RETURN

ON SCREEN: FLAS Inserted

**PRESS** any key



**TYPE** I LIGHT 25 and **PRESS** RETURN

ON SCREEN: LIGH Inserted

**PRESS** any key



**TYPE** I ON 26 and **PRESS** RETURN

ON SCREEN: ON Inserted

**PRESS** any key



TYPE L OFF 27 and **PRESS** RETURN

ON SCREEN: **OFF** Inserted

**PRESS** any key



TYPE: I LOSE 56 and **PRESS** RETURN

ON SCREEN: **LOSE** Inserted

**PRESS** any key



TYPE: I THIRST 66 and **PRESS** RETURN

ON SCREEN: **THIR** Inserted

**PRESS** any key

**NOTE:** "FLAS" and "LIGH" are synonyms



To see the entries you've inserted:

**ENTER** P and **PRESS** RETURN



To return to the Main Menu:

**PRESS** any key, **ENTER** Z and **PRESS** RETURN

## THE MOVEMENT TABLE

As we mentioned earlier, in this phase of our adventure we want the attic to be dark. To do this, **flag 0** must be set upon entering the attic and cleared upon exit. This must be done in the Vocabulary Action Table.

However, the Interpreter checks the Movement Table before it checks the Vocabulary Action Table and if an entry causes movement it will not check the Vocabulary Action Table.

Therefore, we must delete all entries causing movements to and from the attic. To do this from the Main menu:



**ENTER** D and **PRESS** RETURN

MOVEMENT TABLE	
** Format **	** Description **
A <u>loc</u>	Alter an entry
P <u>loc</u> <u>optional</u>	Print on Screen
L <u>loc</u> <u>optional</u>	List to Printer
Z	Return to Main menu



**TYPE** A 0 and **PRESS** RETURN

ON SCREEN: D 1 W 2 U 3 NE 4 E 5 ATT 3 CLOS 2 CELL 1  
KITC 5 BEDR 4



**POSITION** the cursor to the immediate right of "ATT 3" and **DELETE** it by pressing the back space key.



**DELETE** "U 3" using the same procedure



**PRESS** RETURN

ON SCREEN: Altered

You must delete all entries from the attic so:



**TYPE** A 3 and **PRESS** RETURN

ON SCREEN: D 0 LIV 0



**PRESS** SHIFT and CLEAR, then **PRESS** RETURN

ON SCREEN: Altered

**PRESS** any key



To see the changes you've made:

**ENTER** P and **PRESS** RETURN



To return to the Main menu:

**ENTER** Z and **PRESS** RETURN



## MESSAGE TEXT

You need to add a couple more messages in this phase of your adventure to indicate to the player when he is stricken by intense thirst, and when he is dying of thirst. To do this from the Main menu:



ENTER B and PRESS RETURN

MESSAGE TEXT	
** Format **	** Description **
I <input type="text" value="msg"/>	Insert a msg
A <input type="text" value="msg optional"/>	Alter a msg
P <input type="text" value="msg optional"/>	Print on Screen
L	List to Printer
Z	Return to Main menu



ENTER I and PRESS RETURN  
ON SCREEN:



TYPE I'm very thirsty. I need a drink soon or I'll die.  
PRESS RETURN  
ON SCREEN:

PRESS any key



ENTER I and PRESS RETURN  
ON SCREEN:



TYPE I'm dying of thirst.  
PRESS RETURN  
ON SCREEN:



PRESS any key



ENTER I and PRESS RETURN  
ON SCREEN:



TYPE Too bad. You lose.

PRESS RETURN

ON SCREEN: Inserted



PRESS any key



To see the Message Text:

ENTER P and PRESS RETURN



To return to the Main Menu:

ENTER Z and PRESS RETURN

## THE VOCABULARY ACTION TABLE

At this stage of the adventure there are several entries to be inserted in the **vocabulary action table**. Let's begin with entries to make the attic dark. We'll use the AT condition and the SET, GOTO and DESC actions. These are the entries which replace the ones deleted from the movement table.

**NOTE:** When flag 0 is equal to zero it is **light**, when flag 0 is not equal to zero it is **dark**. The action SET, sets a flag equal to 255. The action CLEAR, clears a flag to 0.

When the player is AT the living room (location 0) and moves UP to the attic (location 3) we want to SET flag 0 (making it dark), and describe (DESC) the location.

From the Main Menu:

ENTER G and PRESS RETURN

VOCABULARY ACTION TABLE			
** Format **		** Description **	
A	<u>word1</u> <u>word2</u>		Alter entry
I	<u>word1</u> <u>word2</u>		Insert entry
P	<u>word1</u> <u>word2</u> opt		Print on Screen
L	<u>word1</u> <u>word2</u> opt		List to Printer
Z			Return to Main menu

TYPE I U \* and PRESS RETURN

TYPE AT 0 SET 0 GOTO 3 DESC and PRESS RETURN

ON SCREEN: Inserted

PRESS any key

TYPE I ATT I \* and PRESS RETURN

TYPE AT 0 SET 0 GOTO 3 DESC and PRESS RETURN

ON SCREEN: Inserted

PRESS any key

When the player is AT the attic (location 3) and moves DOWN to the living room (location 0), we want to CLEAR flag 0 (making it light) and DESC (describe) the location.

TYPE I D \* and PRESS RETURN

TYPE AT 3 CLEAR 0 GOTO 0 DESC and PRESS RETURN

ON SCREEN: Inserted

PRESS any key

TYPE I LIV I \* and PRESS RETURN



**TYPE** AT 3 CLEAR 0 GOTO 0 DESC and **PRESS RETURN**  
ON SCREEN: **Inserted**  
**PRESS** any key

**NOTE:** When flag 0 is not equal to 0 and object 0 is not present, the system message "Everything is dark. I can't see." is printed.

In our adventure the player has to be carrying a flashlight that is turned off (object 3) in order turn it on. He also needs the flashlight that is turned on (object 0) to be able to see in the attic (location 3). If you remember, the "on" flashlight is not created yet. We'll use the action SWAP to exchange the positions of objects 0 and 3. This will effectively turn the flashlight on or off.



**TYPE** I FLAS ON and **PRESS RETURN**



**TYPE** CARRIED 3 SWAP 3 0 OK and **PRESS RETURN**  
ON SCREEN: **Inserted**



**TYPE** I ON FLAS and **PRESS RETURN**



**TYPE** CARRIED 3 SWAP 3 0 OK and **PRESS RETURN**  
ON SCREEN: **Inserted**



**TYPE** I FLAS OFF and **PRESS RETURN**



**TYPE** CARRIED 0 SWAP 3 0 OK and **PRESS RETURN**  
ON SCREEN: **Inserted**



**TYPE** I OFF FLAS and **PRESS RETURN**



**TYPE** CARRIED 0 SWAP 3 0 OK and **PRESS RETURN**  
ON SCREEN: **Inserted**

Notice that we inserted two entries to turn on the flashlight and two entries to turn the flashlight off. This was done so that the Interpreter can understand both of the following commands, and so that the player can enter either of these commands and achieve the same result:

TURN ON THE FLASHLIGHT  
TURN THE FLASHLIGHT ON

In this adventure the player scores 50% for opening the safe. If he closes the safe and opens it again, we don't want him to score again so we'll set a flag (12) the first time he opens the safe. All flags are set to 0 when the adventure begins. Consequently, before the safe is opened, flag 12 will be zero. We also don't want to be able to open a safe that is already open, so we'll check for a closed safe.

We'll need two entries to open the safe. Therefore, we must alter the OPEN SAFE entry that is already present to check flag 12 (to determine if the safe has been opened before) and to set flag 12 the first time the safe is opened.



**TYPE** A OPEN SAFE and **PRESS** RETURN

ON SCREEN: PRESENT 7 CARRIED 6 DESTROY 7 CREATE 5  
CREATE 2 LET 30 50 DESC



**MOVE** the cursor to the immediate left of "DESTROY 7"  
and **TYPE** ZERO 12

ON SCREEN: PRESENT 7 CARRIED 6 ZERO 12 DESTROY 7  
CREATE 5 CREATE 2 LET 30 50 DESC



**MOVE** the cursor to the immediate left of "LET 30 50"  
and **TYPE** SET12



**PRESS** RETURN

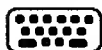
ON SCREEN: Altered



**PRESS** any key



**TYPE** 1 OPEN SAFE and **PRESS** RETURN



**TYPE** PRESENT 7 CARRIED 6 DESTROY 7 CREATE 5 OK



**PRESS** RETURN

ON SCREEN: Inserted



**PRESS** any key



**TYPE** P OPEN SAFE and **PRESS** RETURN to see the changes you made

**NOTE:** If the safe has been **opened**, flag 12 is not equal to 0 and the interpreter will "fall thru" and check the second OPEN SAFE entry. The second entry checks for a key and a closed safe, but does not give any points for opening the safe.

We need to insert an entry to allow the player to close a safe that he has opened. All we have to do is check that an open safe is present and if so, destroy it and create a closed safe.



**TYPE** 1 CLOSE SAFE and **PRESS** RETURN



**TYPE** PRESENT 5 DESTROY 5 CREATE 7 OK



**PRESS** RETURN

ON SCREEN: Inserted



**PRESS** any key

Later, in the Status Table we will set some conditions to cause the player to get **thirsty**. If his thirst isn't quenched, he will die. You will set flag 11 to indicate that the player's thirst has been quenched. An entry in the Status table will check flag 11 to determine if the player has drunk the wine.



TYPE A DRINK WINE and PRESS RETURN  
ON SCREEN: CARRIED 4 MESSAGE 1 OK



**MOVE** the cursor to the immediate left of "OK"  
and TYPE SET 11



PRESS RETURN

ON SCREEN: Altered



**PRESS** any key

Now you must insert the entries that will enable the player to GET and DROP the flashlight. You will make two entries for each:



TYPE I GET FLAS and PRESS RETURN



TYPE PRESENT 3 GET 3 OK and PRESS RETURN

ON SCREEN: Inserted



**PRESS** any key



TYPE I GET FLAS and PRESS RETURN



TYPE GET 0 OK and PRESS RETURN

ON SCREEN: Inserted



**PRESS** any key

Notice that the first entry checks for the presence of the turned off flashlight (object 3). If it is present, the Interpreter gets it, and the action "OK" causes an exit from the table. If the turned off flashlight is not present, the Interpreter goes to the next entry and gets the turned on flashlight (object 0).



TYPE I DROP FLAS and PRESS RETURN



TYPE PRESENT 3 DROP 3 OK and PRESS RETURN

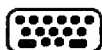
ON SCREEN: Inserted



**PRESS** any key



TYPE I DROP FLAS and PRESS RETURN



TYPE DROP O OK and **PRESS** RETURN  
ON SCREEN: **Inserted**



**PRESS** any key

Notice that the entries for "DROP FLAS" have the same conditions and format as those for "GET FLAS." They work in the same manner.

To see the entries:

**ENTER** P and **PRESS** RETURN



U	*	Conds	AT	0	
		Acts	SET	0	
			GOTO	3	
			DESC		
D	*	Conds	AT	3	
		Acts	CLEAR	0	
			GOTO	0	
			DESC		
FLAS	ON	Conds	CARRIED	3	
		Acts	SWAP	3	0
			OK		
FLAS	OFF	Conds	CARRIED	0	
		Acts	SWAP	3	0
			OK		
ON	FLAS	Conds	CARRIED	3	
		Acts	SWAP	3	0
			OK		
OFF	FLAS	Conds	CARRIED	0	
		Acts	SWAP	3	0
			OK		
LIVI	*	Conds	AT	3	
		Acts	CLEAR	0	
More ...					

This is the first of 5 screens. To see the entire listing, continue to press RETURN after each screen is displayed.

To return to Main Menu:

**ENTER** Z and **PRESS** RETURN





## THE STATUS TABLE

During this part of our adventure we want the player to **lose** after taking 21 turns without returning to the living room with the jewel. The "FINI\*" entry will check the conditions: **AT 0 & PRESENT 2** (in the living room without the jewel). As long as these conditions are not satisfied, the next entry in the table will be checked. Make the following entry:

From the Main Menu:

ENTER H and PRESS RETURN

STATUS		
** Format **	** Description **	
A <u>word1</u> <u>word2</u>	Alter entry	
I <u>word1</u> <u>word2</u>	Insert entry	
P <u>word1</u> <u>word2</u> opt	Print on Screen	
L <u>word1</u> <u>word2</u> opt	List to Printer	
Z	Return to Main menu	

TYPE I LOSE \* and PRESS RETURN

**NOTE:** The **word value** of LOSE is greater than that of FINI to insure that it will follow the FINI \* entry.

TYPE: GT 31 20 ZERO 32 PAUSE 70 MESSAGE 5 SOUND  
0 175 10 10 PAUSE 25 SOUND 0 225 10 10 PAUSE  
25 SOUND 0 250 10 10 PAUSE 25 SOUND 0 50 2  
15 PAUSE 35 SOUND 0 0 0 0 SCORE TURNS END

PRESS RETURN

ON SCREEN: I LOSE \* Inserted

PRESS any key

This entry limits the player to 21 turns. After 21 turns the message **"Too bad. You lose."** is printed and a losing sound is made. The score and number of turns are displayed and the player is asked if he wants to play again.

**NOTE:** Flags 31 and 32 hold the low byte and high byte of the number of turns.

We are going to make one more entry in the Status Table to make our adventure more exciting. We want the player to get thirsty after 7 turns, remain thirsty for 5 turns and then die following the next turn. Drinking the wine will quench his thirst and keep him alive. To do this we have to set some flags and check them.

Flag 11 will be set when the player drinks the wine, indicating that his thirst was quenched.

**NOTE:** Flags 5-8 automatically decrement each turn.

Flag 5 will be set to 6 on the seventh turn. When **flag 5 = 1** the player will die.

Three THIRST \* entries must be made in the status table for all this to work properly. The first entry will set flag 5 after the seventh turn:



TYPE 1 THIRST \* and PRESS RETURN



TYPE EQ 31 7 ZERO 32 LET 5 6 and PRESS RETURN

ON SCREEN: 1 THIRST \* inserted

PRESS any key



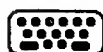
TYPE 1 THIRST \* and PRESS RETURN



TYPE GT 5 1 ZERO 11 MESSAGE 3 and PRESS RETURN

ON SCREEN: 1 THIRST \* inserted

PRESS any key



TYPE 1 THIRST \* and PRESS RETURN

Notes: Press 5 = 1. The program will die



TYPE: EQ 5 1 ZERO 11 MESSAGE 4 PAUSE 100 SOUND 0  
175 10 10 PAUSE 25 SOUND 0 225 10 10 PAUSE  
25 SOUND 0 250 10 10 PAUSE 25 SOUND 0 50 2  
15 PAUSE 35 SOUND 0 0 0 0 SCORE TURNS END



PRESS RETURN

ON SCREEN: I THIRST \* Inserted



PRESS any key

To see the entry:



TYPE P THIR \* and PRESS RETURN



THIR *	Conds	EQ	31	7			
		ZERO	32				
	Acts	LET	5	6			
THIR *	Conds	GT	5	1			
		ZERO	11				
	Acts	MESSAGE	3				
THIR *	Conds	EQ	5	1			
		ZERO	11				
	Acts	MESSAGE	4				
		PAUSE	100				
		SOUND	0	175	10	10	
		PAUSE	25				
		SOUND	0	225	10	10	
		PAUSE	25				
		SOUND	0	250	10	10	
		PAUSE	25				
		SOUND	0	50	2	15	
		PAUSE	35				
		SOUND	0	0	0	0	
		SCORE					
		TURNS					

More ...

If your entries do not match the ones shown above, retype as needed.

Notice that the second "THIR \*" entry prints the message **"I'm very thirsty. I need a drink soon or I'll die"**. This message gets printed when flag 5 is greater than 1 and flag 11 is equal to 0, indicating that the wine has not been drunk. The conditions for the third entry are satisfied when flag 5 is equal to 1 and flag 11 has not been set. (Flag 11 gets set when the wine is drunk).

## Number of Portable Objects

The last change we'll make to our adventure is to **limit** the **number of objects** that the player can carry. This will make the adventure more difficult and thus, more challenging to solve. The procedure is very simple. From the Main menu:



**ENTER N** and **PRESS RETURN**

ON SCREEN: Enter (0-255) and press RETURN



**TYPE 2** and **PRESS RETURN**

**PRESS** any key

This entry will not permit the player to carry any more than two objects (not including any objects that are worn).

## Save the Database

At this point, you have entered all the database additions for Phase III of your adventure. You **saved** these changes after each phase as separate databases. From the Main menu:



**ENTER I** and **PRESS RETURN**

ON SCREEN: Type in name of file



**TYPE PHASE3** and **PRESS RETURN**

When the database is saved correctly, you will be prompted to press any key to continue. If an error message is displayed, resave the database until it is saved correctly.

## Test the Adventure

Now it is time to **test** the adventure. From the Main menu:



**ENTER K** and **PRESS RETURN**

ON SCREEN: Do you want to see diagnostic flags?



## ENTER Y and PRESS RETURN

The screen displays a description of location 0 and the starting values of all the flags. As you move from room to room, and GET and DROP objects, you will see the values of the flags change. Flag 31 (the turns count) will start at 1 and be incremented every turn. When you are in the living room, **TYPE** in the following commands:

SOUTH	will reply	I can't go in that direction.
GET JEWEL	will reply	It's not here.
OPEN SAFE	will reply	I can't.
ATTIC	will reply	Everything is dark. I can't see.

At any time while testing this adventure you may **exit** by quitting:



## TYPE QUIT and PRESS RETURN

ON SCREEN: Do you really want to quit now?



TYPE Y (for yes) and PRESS RETURN



VIEW:

```

Tell me what to do.
> QUIT
Do you really want to quit now?
> Y
You have taken * turns.
You have scored * %

                END OF GAME

Do you want to try again?
> N
Bye. Have a nice day.

Ok
  
```



TYPE N

This is the standard procedure for exiting an adventure. If the player decides to interrupt the game, he may exit at any time by typing **quit** or **stop**.

**Winning and losing** are other ways of exiting. If the player wins, he will see the message: "You win!!! Congratulations!!!" (or an alternate win-message the designer enters in the message text table). Losing displays the message: "Too bad, you lose". In either case, you are given the opportunity to try again.

## Save the Adventure

You have nearly completed Phase III of AdventureWriter, but you must take one more crucial step before becoming the proud owner of your very own stand-alone program. You must **save** your adventure.

Make sure that your application disk (and not the Adventure-Writer Disk) is in the disk drive.

From the Main menu:

**ENTER** L and **PRESS** RETURN

ON SCREEN: **--- SAVE ADVENTURE ---**

ON SCREEN: **Type in name of file**

**TYPE** THIEF and **PRESS** RETURN

When the adventure has been saved correctly, you will be prompted to press any key to continue. If an error message is displayed, resave the adventure until it is saved correctly.

You have just completed your first adventure!

We highly recommend that you read the reference section of this manual before attempting to create your own adventure. You will avoid perils and pitfalls of your own by doing so.



To run your adventure program:

**REMOVE** all cartridges from the slots

**POWER UP** System:

- Turn on disk drive
- Turn on monitor or TV

**INSERT** application disk containing your created adventure into the disk drive then Turn on computer

From the DOS menu:

**ENTER** L and **PRESS** RETURN

**TYPE** THIEF.ADV and **PRESS** RETURN

Your program will then load and start to run.





## **DETAILED DESCRIPTION OF THE DATABASE**

The database consists of a number of inter-related tables and an area of miscellaneous information (eg: Number of portable objects). The tables present are:

### **A The Vocabulary Text Table**

Each entry in the table uses 5 characters and contains a word ( or the first four characters, if the word is longer than four characters ) and a word value in the range 1-254. Words with the same word value are called synonyms. The entries are held in ascending order of word value and within each word value, entries with the least amount of characters come first.

eg:           U  
              UP  
              CLIM  
              ASCE

Where entries with the same word value also have the same number of spaces, the entry inserted first comes earlier ( eg: CLIM[B] was inserted before ASCE[ND] ).

Note 1. When the Editor has to convert a word value to a word, it takes the first word with that value.

Note 2. Word values less than 13 should be reserved for movement words.

### **B The Message Text Table**

This table contains the text of any messages which are needed for the adventure. The messages are numbered from 0 upwards and each one uses 3 characters plus the length of the text.

## **C The Location Descriptions Table**

This table, which has an entry for each location, contains the text which is printed when a location is described. Each entry uses 3 characters plus the length of the text. The entries are numbered from 0 upwards and location 0 is the location at which the adventure starts. Whenever a new location is inserted, an empty (null) entry for that location is also made in the Movement table.

## **D The Movement Table**

This table has an entry for each location and each entry may either be empty (null) or contain a number of "movement pairs". A movement pair consists of a word value in the vocabulary followed by a location number, and means that any word with that word value causes movement to that location. A typical entry could be SOUTH 6 EAST 7 RETURN 6 NORTH 5. This means that SOUTH or RETURN or their synonyms cause movement to location 6, EAST or its synonyms to location 7 and NORTH or its synonyms to location 5. Each entry uses 3 characters plus 2 characters for each movement pair.

**Note 1.** The movement pairs contain the word value, not the actual word, and if a word value is deleted from the vocabulary, then all movement pairs which contain that word value are also deleted.

**Note 2.** When you play the adventure, only the first recognized word (W1) will cause movement.

**Note 3.** If any movements are to be performed in the Event or Status tables using the action GOTO then exclude those movements from the Movement table.

## **E The Object Descriptions Table**

This table, which has an entry for each object, contains the text which is printed when an object is described. Each entry uses 3 characters plus the length of the text.

An object is anything in the adventure which may be manipulated and is numbered from 0 upwards. Object 0 is assumed by the Interpreter to be a source of light such as a flashlight or torch. Whenever a new object text is inserted, an entry of "not created" is made for that object in the Object Starting Location table.

## **F The Object Starting Location Table**

This table has a one character entry for each object, which specifies the location at which the object is situated at the beginning of the adventure. An object can also start the adventure being worn, carried or not created.

## **G The Vocabulary Action Table**

This table (together with the Status table) forms the main part of the database. Each entry contains 2 word values followed by any number of conditions and then (normally) at least one action. When you play the adventure, if there is an entry in the table with the word values entered and having the conditions specified satisfied, then the actions are performed. The conditions and actions that may be present and the effect that they have are fully specified in the description by the Interpreter. The order of entries in the table is in ascending order of the first word value. Entries which have the same first word value are held in ascending order of the second word value. Entries with the same first and second word values are held in the order they were inserted into the database (ie: they must be inserted in the order required). An example of the order of the table, with word values shown in brackets is as follows:-

NOTE: word values (shown in brackets) will appear only in the Vocabulary Text table	LOOK	[30]	UP	[9]
	LOOK	[30]	DOWN	[10]
	LOOK	[30]	*	[255]
	GET	[100]	KEY	[16]
	GET	[100]	LAMP	[26]

Each entry in the table has an overhead of 6 characters and each condition and action uses 1, 2, or 3 characters depending on the number of parameters specified.

Note 1. If a word value is deleted from the Vocabulary Text Table, then all entries in the Event and Status tables which contain that word value are deleted.

## H The Status Table

This table has exactly the same format as the Vocabulary Action table. When you play the adventure the Status table is scanned between turns to see if the adventure wants anything to happen. The Vocabulary Action table can be considered as the player's table and contains entries which are dependent on the words entered, while the Status table is the computer's table and contains entries which are independent of the words entered by the player. The words in the Status table can, however, be used to position entries at the required place and/or as a reminder of the purpose of the entries.

## I The AdventureWriter Message Table

This table contains the messages used by the Interpreter. Each entry uses 3 characters plus the length of the text. The description of the Interpreter shows how these messages are used.

eg:        **I'm hungry**  
             **I'm dying of sun-stroke**  
             **I need a drink**    etc ...

## DETAILED DESCRIPTION OF THE EDITOR

Within this section certain abbreviations will be used for the sake of simplicity:

**Abbreviations:** message number ..... **msg** \* (0-254)  
                         location number ..... **loc** \* (0-251)  
                         object number ..... **obj** \* (0-254)

## A Vocabulary

Words may be inserted or deleted; the synonyms of a word may be displayed, or the vocabulary may be printed:

To INSERT a word



Type: **I word word\*** (word\* is between 1 & 254 and is assigned as the value of the word)

If **word** is not already present in the vocabulary it is inserted with a word value specified above.

To DELETE a word



Type: **D word**

If the variable **word** is present in the vocabulary, it and its word value are deleted. If synonyms of the word deleted are present in the vocabulary, no further action is taken. However, if no synonyms are present, then:

- a) all entries in the Vocabulary Action and Status tables which use this word value are also deleted.
- b) all movement in the Movement table which use this word value are also deleted.

To SHOW SYNONYMS



Type: **S word**

If **word** is present in the vocabulary, it and all other words with the same word value are displayed.

To LIST to printer or PRINT on screen



Type: **L** (to list to printer) or **P** (to print on screen)

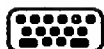
Points to note:

- a) Be careful in using delete as it can also affect the Vocabulary Action, Status and Movement tables.
- b) Words with a word value of less than 13 are assumed to be movement words by the Interpreter and can cause the message "I can't go in that direction" to be printed instead of "I can't".
- c) When entries in the Vocabulary Action, Status and Movement tables are affected, the Editor may take a few minutes to delete a word.

## B Message Text

Message texts may be inserted, amended/alterd or printed:-

To INSERT



Type: **I**, hit **RETURN**, enter message

The next available message number is used and a null (empty) entry is made for it in the Message Text table. An automatic call to the amend/alter routine is then made to allow the user to amend/alter the null entry.

To AMEND/ALTER



Type: **A msg #**

The Editor will cause the existing text for the message **msg #** to be copied to the input buffer (input storage area) and displayed on the screen for amending/altering. Pressing **RETURN** will cause the existing text to be replaced with the contents of the input buffer.

To LIST to the printer or PRINT to the screen



Type: **L msg #** to list to printer (**msg#** is optional)  
or **P msg #** to print to screen (**msg#** is optional)

Printing starts with the text for message msg\* or at the beginning Message Text table if msg\* is not specified.

Note: There is a limit of 255 messages.

## C Location Descriptions

Location texts may be inserted, amended/alterd, or printed:-

To INSERT



Type: I, hit **RETURN**, enter text

The next available location number is automatically used and a null entry is made for it in both the Movement and Location Text tables. Processing then continues with an automatic call to the amend routine to allow the user to amend the null entry already set up in the Location Text table.

To AMEND/ALTER



Type: **A loc #**

The existing text for Location loc # is copied to the input buffer and displayed on the screen for amending/altering. When RETURN is pressed, the existing entry is replaced with the contents of the input buffer.

To LIST to printer or to PRINT on screen



Type: **L loc #** to list to printer (loc # is optional)  
or **P loc #** to print on screen (loc # is optional)

Printing starts with the text for location loc # or , if loc # is not specified, at the beginning.

Points to note:-

- The start of an adventure is always at location 0
- There is a limit of 252 locations.

## D Movement Table

Movements may be amended/alterd or printed:-

To AMEND/ALTER

Type: **A** loc \*



The existing entry for location loc \* is decoded, copied to the input buffer and displayed on the screen for amending/altering. When RETURN is pressed the input buffer is assumed to be empty or to contain any number of movements in the format (**word** loc \*). **Word** must be present in the Vocabulary Text table and loc \* must be present in the Location Description table. If there are no syntax errors, the existing entry is replaced with an encoded copy of the input buffer (ie: words changed to word values).

To LIST to printer or to PRINT on screen

Type: **L** loc \* to list to printer (loc \* is optional)

or **P** loc \* to print on screen (loc \* is optional)



Printing starts with the entry for location loc \* or at the beginning if loc \* is not specified.

Points to note:

- a) A location text must be present for a location before movements can be present.
- b) Any words in the Vocabulary may be used in the Movement table.
- c) When an entry is decoded (for Amend/Alter or Print), the word value is changed into the first word in the Vocabulary with that word value.



## E Object Descriptions

Object text may be inserted, amended/alterred, or printed:-

To INSERT



Type: **I**, hit **RETURN**, enter text

The next available object number is used and an empty (null) entry is made for it in the Object Text table. An entry of "not created" is also made for it in the Object Starting Location table. Processing then continues with an automatic call to the amend/alter routine to allow you to amend/alter the null entry already set up in the Object Text table.

To AMEND/ALTER



Type: **A** **obj \***

The existing text for the object **obj \*** is copied by the Editor to the input buffer and displayed on the screen for amending/altering. Pressing **RETURN** will replace the existing text with the contents of the input buffer.

To LIST to printer or PRINT to screen



Type: **L** **obj \*** to list to printer (**obj \*** is optional)  
or **P** **obj \*** to print on screen (**obj \*** is optional)

Printing starts with the text for object **obj \*** or at the beginning if **obj \*** is not specified.

Points to note:

- a) Object 0 is always considered by the Interpreter to be a source of light. (eg: candles, flashlights, torches, etc.)
- b) There is a limit of 255 objects in an adventure.

## F Object Starting Locations

The location at which an object is situated at the start of the adventure may be amended/alterd or the Object Start Location table may be printed:-

To AMEND/ALTER

Type: **A obj \* loc \***

The existing entry for object obj \* is replaced with loc \* which must be present either in the Location Text table or be one of the special location numbers.

(ie: 252 not created - 253 worn - 254 carried)



To LIST to printer or to PRINT on screen

Type: **L** to list to printer

or **P** to print on screen

There are no other parameters needed to LIST or PRINT

An object text must be present for an object before its starting location can be present.

## G Vocabulary Action Table

Entries may be inserted, amended/alterd, deleted or printed:-



To INSERT

Type: **I word1 word2**

**Word1** and **word2** MUST be asterisks (\*) or words which are in the vocabulary. The word values of **word1** and **word2** (any asterisk has a value of 255) are used to find the correct place in the table for the new entry to be created. If any entries already exist for **word1** or **word2** then the new entry will be created after the previous entries. An empty (null) entry is created at the appropriate place and an automatic call is made to the amend/alter routine to allow you to amend/alter the empty (null) entry.

To AMEND/ALTER



Type: **A word1 word2**

The first entry in the table with word values of **word1** and **word2** is first copied to the input buffer and then will be displayed on the screen for amending/altering.

When RETURN is pressed the input buffer is assumed to be empty (in which case the existing entry is deleted), or to contain number of valid conditions, followed by at least one valid action. If there are no syntax errors, the existing entry is replaced with the contents of the input buffer. Any following entries in the table with the same word values (ie: **word1** and **word2**) are then displayed in turn for amending/altering in the same way.

The following tables contain all valid AdventureWriter Conditions and Actions. These entries are used in building the Vocabulary Action and Status tables.

### CONDITIONS FORMAT

This particular table presents the required structure for conditions, along with an explanation of actions and objects needed (or not needed) for each condition satisfied.

loc\* = specified location number    obj\* = specified object number

CONDITION	EXPLANATION
AT loc*	Satisfied if the player is <b>at</b> specified location number
NOTAT loc*	Satisfied if the player is <b>not at</b> specified location number
ATGT loc*	Satisfied if the player is <b>at</b> a location whose number is <b>greater than</b> specified location number
ATLT loc*	Satisfied if the player is <b>at</b> a location whose number is <b>less than</b> specified location number
PRESENT obj*	Satisfied if the specified object is <b>carried, worn</b> or <b>at</b> the current location
ABSENT obj*	Satisfied if the specified object is <b>not carried, not worn</b> or <b>not at</b> the current location
WORN obj*	Satisfied if the specified object is <b>worn</b>
NOTWORN obj*	Satisfied if the specified object is <b>not worn</b>

CONDITION	EXPLANATION
CARRIED obj*	Satisfied if the specified object is currently <b>carried</b>
NOTCARR obj*	Satisfied if the specified object is currently <b>not carried</b>
CHANCE percent	Satisfied if <b>percent</b> is <b>greater than</b> a random number between 1 and 100 created by the computer.
ZERO flg*	Satisfied if the flag specified by <b>flg*</b> is <b>equal</b> to zero
NOTZERO flg*	Satisfied if the flag specified by <b>flg*</b> is <b>not equal</b> to zero
EQ flg* value	Satisfied if the flag specified by <b>flg*</b> is <b>equal</b> to the specified value
GT flg* value	Satisfied if the flag specified by <b>flg*</b> is <b>greater than</b> the specified value
LT flg* value	Satisfied if the flag specified by <b>flg*</b> is <b>less than</b> the specified value

## ACTION FORMAT

Read this table to understand what actions may be incorporated into your adventure.

loc\* = location number flg\* = flag number obj\* = object number

<b>ACTION</b>		<b>EXIT</b>	<b>DESCRIPTION</b>
<b>ANYKEY</b>		No	Prints "Press any key to continue " and continues when a key is pressed
<b>BORDER</b>	value	No	Changes current screen color to specified color
<b>CLEAR</b>	flg*	No	The specified flag number is set to zero
<b>CLS</b>		No	Clears the screen
<b>CREATE</b>	obj*	No	Changes the object location from "not created" to current location number
<b>DESC</b>		Always	Prints current location description
<b>DESTROY</b>	obj*	No	Changes specified object to "not created"
<b>DONE</b>		Always	Exits from the table
<b>DROP</b>	obj*	May	Causes a carried or worn object to be dropped .
<b>DROPALL</b>		No	Causes all carried or worn objects to be dropped
<b>END</b>		Always	Ends the adventure
<b>GET</b>	obj*	May	Gets the specified object from the current location

<b>GOTO</b>	loc*	No	Changes current location to specified new location number
<b>INVEN</b>		Always	The list of objects currently carried is printed
<b>LET</b>	flg* value	No	The value of the flag specified is changed to the value specified
<b>LOAD</b>		Always	Loads a previously saved adventure with current position disk
<b>MINUS</b>	flg* value	No	The flag number specified is reduced by the value specified
<b>MESSAGE</b>	msg*	No	Prints the specified message
<b>OK</b>		Always	Prints "OK"
<b>PAUSE</b>	value	No	Pauses adventure for value/50 seconds
<b>PLACE</b>	obj* loc*	No	Puts specified object into specified location
<b>PLUS</b>	flg* value	No	The specified flag is incremented by the specified value
<b>QUIT</b>		May	Causes the question "Do you really want to quit now ?" to be displayed. If player response is "no" then the adventure will continue

<b>REMOVE</b>	obj*	May	Changes a worn object into a carried object
<b>SAVE</b>		Always	Saves current position to disk or tape while running an adventure
<b>SCORE</b>		No	Prints player's current score
<b>SCREEN</b>	value	No	Changes current screen color to specified color
<b>SET</b>	flg*	No	Sets specified flag to 255
<b>SOUND</b>	v p d vol		Similar to Atari BASIC SOUND statement. Sets voice, pitch, distortion and volume. See pp. 103-104.
<b>SWAP</b>	obj* obj*	No	The locations of the specified objects are inter-changed
<b>TEXT</b>	value	No	Changes the current text to specified intensity.
<b>URNS</b>		No	Prints the number of turns a player has taken
<b>WEAR</b>	obj*	May	Causes an object that is currently carried to be worn



## Editor Error Messages

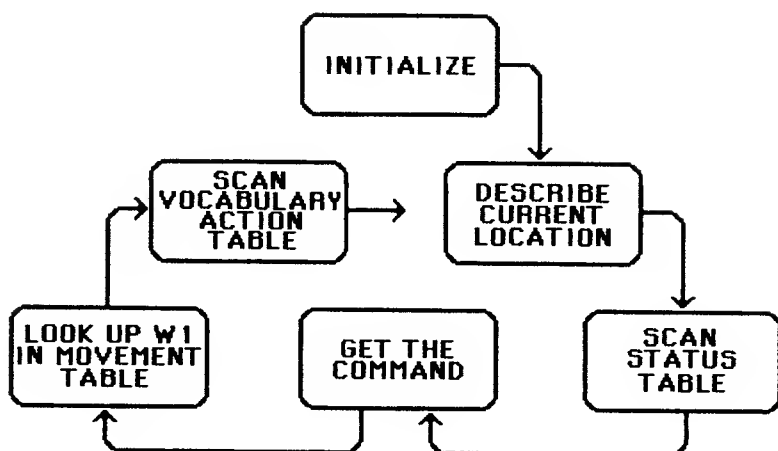
<u>Error</u>	<u>Meaning</u>
I/O ERROR	Disk error. Note that an I/O ERROR during a LOAD means that the database is corrupt or the file is not found. Make sure you have the correct disk and reload the database until it is loaded successfully.
DATABASE FULL	There is not enough room in the database for what you are trying to do.
LIMIT REACHED	The maximum number of locations, messages or objects is already present.
INPUT BUFFER FULL	The entry for Vocabulary Action, Status or Movement table is too large for the input buffer (very unlikely to occur). Shorten the entry.

NOTE: If an abnormally large entry is inserted in the movement table using abbreviations, (N 1 S 2 E 3 ...) and the abbreviations are deleted from the vocabulary, the movement entry, when decoded (NORT 1 SOUT 2 EAST 3 ...) could be too big for the input buffer. If this happens, an out of memory error will occur. To remedy this, reinsert the abbreviations in the vocabulary.

## The Interpreter

The Interpreter controls the flow of the adventure. It reads what the player types in and tries to execute his commands. When a command is not understood, the Interpreter responds with a message saying that it does not understand. If a command cannot be executed, the Interpreter will respond with "I can't", or another more specific message.

The diagram below illustrates the program flow:



### **Initialize -**

All flags are set to zero except for flag 1 which is set to the number of objects initially carried.

### **Describe Current Location -**

The flags are examined by the Interpreter to determine what to print. The current location description is printed when the location is light or when object 0 is present.

If it is dark and object 0 is absent, the message "Everything is dark. I can't see." is printed.

### **Scan the Status Table -**

The first entry in the Status table is examined, by the Interpreter; if the conditions are satisfied, the action(s) that follow will be performed: if not satisfied, the next entry in the table will be examined. This procedure repeats for each entry in the table or until an exit occurs.

### **Get the Command -**

The Interpreter looks up the first four letters in the vocabulary text table and the values of the first two words found are stored in variables W1 and W2.

### **Look up W1 in Movement table -**

The Interpreter searches the movement table for the current location to see if the word value in W1 is present. If it is, the current location is set to the value following W1 in the movement table.

### **Scan the Vocabulary Action Table -**

The Interpreter looks at each entry in the table until it finds one that starts with (W1 and W2). When it finds a matching entry, conditions are checked and when satisfied, the actions are performed. This procedure is repeated for the remaining entries in the table or until an exit occurs.

The basic structure of the Interpreter has its origins in an article written by Ken Reed and published in the August 1980 issue of Practical Computing. Although the Interpreter in AdventurWriter now has little in common with the article, the terms used to describe it are similar to those used in the article.

## Diagnostic Flags

Flags are just variables that hold numbers. The flags are numbered from 0 to 32. Some flags are controlled by the Interpreter and you, the designer, control the others.

Let's try a simple example. Suppose we want to know whether or not the player has gone to the beach before he gets to the dungeon (a 1 in an imaginary Adventure). To do this, we take the following approach:

When the player gets to the beach, make variable 11 equal to 123. That way, when the player gets to the dungeon, all we have to do is look at variable 11. If it is equal to 123, then we know that the player has been to the beach.

Here's how we would do it:

In the Status Table, we check if the player is at location 10, which is the beach:

CHEC *	Conds	AT	10
	Acts	LET	11 123
	:		
	:		

Now, also in the Status Table, we check to see if we're at Location 33 (the dungeon) and if we've been to the beach:

CHK2 *	Conds	AT	33
		EQ	11 123
	Acts		
	:		
	:		

All flags start out with a value of zero until you or the Interpreter changes them. You can change any of the flags but the only ones that you totally control are 11-29. (The Interpreter doesn't have any effect on these). The other flags (0-10, 30-32) are affected in one way or another by the Interpreter. Let's examine each flag in detail:

Flag 0	When you set this flag to zero, the Interpreter assumes that it is light and will display the normal location description. When it is set to any other value, the Interpreter assumes it is dark and prints the message "Everything is dark. I can't see.", as a location description.
Flag 1	Holds the count of the number of objects carried.
Flag 2	Decrementd by 1 or reduced in number each time a location is described.
Flag 3	Decrementd by 1 each time a location is described and it is dark (Flag 0>0).
Flag 4	Decrementd by 1 each time a location is described and it is dark (flag 0>0) and object 0 is absent.
Flags 5-8	Decrementd by one each turn.
Flag 9	Decrementd each turn when it is dark (flag 0>0)
Flag 10	Decrementd each turn when it is dark (flag 0>0) and object 0 is absent.
Flags 11-29	Ordinary flags.
Flag 30	Holds the player's score and is printed when the action SCORE is performed. You enter the value in this flag.
Flag 31	Holds the turns count (low byte).
Flag 32	Holds the turns count (high byte).

NOTE: Flag 31 is incremented every turn. When it reaches 256, flag 32 is incremented and flag 31 is reset to zero. Then the process starts over again.

Let's use some flags in an example.

Suppose we want to print message 1 after the player has taken 10 turns, and then print message 2 after the player takes 7 more turns.

				<u>Explanation</u>
EXAMPLE	*	Conds EQ	31 10	- IF flag 31=10
		ZERO	32	- and if flag 32=0
	Acts	LET	5 7	- THEN set flag 5=7
		MESSAGE	1	- and print message number 1
				- ELSE
EXAMPLE	*	Conds ZERO	5	- IF flag 5=0
	Acts	MESSAGE	2	- THEN print message number 2

NOTE: When you test your adventure, you will have the option of seeing 36 flags displayed directly on your screen. You can see their values, and how and when they change, as you test your adventure. You must answer "yes" to the question "Do you want to see diagnostic flags?" in order to take advantage of this helpful testing aid.

The last three flags that are displayed are in reverse video. The first two hold the values of the two words that the Interpreter understands, (255=word2 does not exist or it is not in the vocabulary). The last flag holds the current location number.

## AdventureWriter Sounds

Sounds are available for you to enhance your adventure further. The information below provides an introduction to possible effects.

The first value after "SOUND" is the voice number (0-3). You may use all four voices simultaneously. The second value is pitch (0-255). The table below gives pitch values for specific notes. The third value, distortion, creates special effects. The fourth value is volume (0-15), where a volume of 0 is used for turning the sound off. Using the PAUSE action with SOUND, many different sound effects can be generated.

NOTE	PITCH
C	29
B	31
A*	33
A	35
G*	37
G	40
F*	42
F	45
E	47
D*	50
D	53
C*	57
C	60
B	64
A*	68
A	72
G*	76
G	81

NOTE	PITCH
F*	85
F	91
E	96
D*	102
D	108
C*	114
C	121
B	128
A*	136
A	144
G*	153
G	162
F*	173
F	182
E	193
D*	204
D	217
C*	230
C	243

For a gun sound:

<b>SOUND</b>	0	90	4	15
<b>PAUSE</b>	5			
<b>SOUND</b>	0	100	4	15
<b>PAUSE</b>	5			
<b>SOUND</b>	0	100	4	7
<b>PAUSE</b>	5			
<b>SOUND</b>	0	100	4	5
<b>PAUSE</b>	5			
<b>SOUND</b>	0	100	4	3
<b>PAUSE</b>	5			
<b>SOUND</b>	0	100	4	1
<b>PAUSE</b>	9			
<b>SOUND</b>	0	0	0	0

For an explosion:

<b>SOUND</b>	0	100	8	15
<b>PAUSE</b>	8			
<b>SOUND</b>	0	100	8	14
<b>PAUSE</b>	8			
<b>SOUND</b>	0	100	8	13
<b>PAUSE</b>	8			
<b>SOUND</b>	0	100	8	12
<b>PAUSE</b>	8			
<b>SOUND</b>	0	100	8	10
<b>PAUSE</b>	9			
<b>SOUND</b>	0	100	8	8
<b>PAUSE</b>	9			
<b>SOUND</b>	0	100	8	6
<b>PAUSE</b>	9			
<b>SOUND</b>	0	100	8	4
<b>PAUSE</b>	9			
<b>SOUND</b>	0	100	8	3
<b>PAUSE</b>	9			
<b>SOUND</b>	0	100	8	1
<b>PAUSE</b>	10			
<b>SOUND</b>	0	0	0	0



## Colors

The following table indicates the numbers which may be used to set the **screen** and **border colors**. Each color has several shades available. The larger the number, the lighter the shade of the color. For example, a **0** would be a dark gray (black) and a **14** would be a light gray (white).

<b>COLOR</b>	<b>NUMBERS</b>
Gray	0-14
Gold	16-18
Orange	32-46
Red-orange	48-62
Pink	64-78
Pink-purple	80-94
Purple-blue	96-110
Blue	112-126
Blue (slightly darker)	128-142
Light-blue	144-158
Turquoise	160-174
Green-blue	176-190
Green	192-206
Yellow-green	208-222
Orange-green	224-238
Light-orange	240-254

## ADVENTUREWRITER CAPACITIES

**Vocabulary** - May be numbered from 1 to 254

**Messages** - May be numbered from 0 to 254

**Location descriptions** - May be numbered from 0 to 251

**Object descriptions** - May be numbered from 0 to 254

If you reach the maximum number of Vocabulary words, Messages, locations, or objects, then a "limit reached" message will be printed.

### Caution:

When creating an actual adventure, it is possible to run out of memory, which would cause an error in operation and likely corrupt your database. Use the option **M** from the Main menu every so often to see how much available memory you have left. For safety's sake, leave yourself plenty of space.

## **Glossary**

- Acts -** The actions performed when all satisfied.
- Conditions -** The conditions necessary for an action to be executed.
- Database -** A large grouping of information, stored in memory or on disk. This database holds the information your adventure uses.
- Editor -** The program that allows you to create or change an adventure.
- Filename -** A name, made by typing letters of your choice, that the AdventureWriter will use to refer to your database or completed adventure. Remember this name. You'll need it to load your database or adventure once it's saved on disk.
- Flag -** A variable capable of holding a number (0-255). Each flag itself is numbered 0-32. Flags are used for keeping track of things such as thirst, hunger, etc.
- Interpreter -** The program that takes the data you have created with the Editor and uses it to run the adventure.

- Inverse** - The effect of a word or character is printed out with a solid background and a negative foreground.
- Menu** - A list on the screen of various options you may take. To choose an option, type one letter that matches the command you want and press RETURN.
- Null entry** - Inserting a RETURN with nothing preceding may put a blank entry into the current table or description. An accidental null entry may be amended : at any time. **Caution:** Your entry number may also be incremented.
- Syntax error** - Indicated by a ' ? '. An error due to misspelling, typing an invalid option, or an incorrect format.

## Main Menu Glossary

### A . . Vocabulary - Text

The words (with their corresponding number) which the program will understand when the adventure is run.

### B . . Message - Text

This table will contain all messages which will be printed; based on conditions met during an adventure.

**eg: I'm dying of starvation!**

### C . . Location - Descriptions

The descriptions printed for each location. These are exact descriptions of each location which are entered at the beginning of the adventure creation.

### D . . Movement - Table

This table contains the relative positions of each location. The inter connections between all locations are entered in this table.

### E . . Object - Descriptions

This table contains the complete descriptive text required for each object. **Important:** Object 0 is **always** considered a source of light (eg: candles, flashlights, etc.). Remember this when creating your adventure.

<b>F . . Object - Starting Locations</b>	This table will contain the exact location of each object at the start of your created adventure. Entries cannot be inserted. When you insert object text for an object, the Editor automatically inserts an entry of " not created " for that object in the table.
<b>G . . Vocabulary - Action Table</b>	The table that you create to tell the Interpreter how to respond when a player types in a command.
<b>H . . Status Table -</b>	This table checks various flags and other information just before the Interpreter asks the player what to do.
<b>I . . Save A - Database</b>	This options allows you to transfer your created database to disk
<b>J . . Verify A - Database</b>	This option allows you to verify that your database was saved without errors.
<b>K . . Load a - Database</b>	This option allows you to load a previously saved database from disk so that you may use the Editor either to examine or to change any information.
<b>L . . Test This - Adventure</b>	Using this option, you may test a partially or fully completed adventure and return to the Editor.

- M . . Save This - Adventure**      You would use this option to save a **fully completed** adventure. To save a partially completed adventure, you must use the option **" I " - Save a Database.**
- N . . Verify This - Adventure**      Use this option to check that the adventure was saved without errors.
- O . . Memory - Available**      This option will display the amount of unused memory left for you adventure.
- P . . # Of Portable - Objects**      With this option you may set the maximum number of objects (torches, keys, food, etc.) a player may carry at one time.
- Q . . Adventure - Writer Messages**      With this option, you may examine or alter AdventureWriter messages. These are pre-set messages which cannot be added to or deleted from.
- ✚ . . Exit Adventure-Writer**      Use of this option will cause the computer to restart.







- Actions 5,54,94
- Adventure 5,42,97,98
- Adventure Map 16
- AdventureWriter
  - Capacities 105
  - Definition 5
  - Message Table 84
  - Objectives 11,16
  - Sound 103
- Amend/Alter 19,22,41,64,69

- Colors 105
- Commands 5
- Conditions 53,55,72,92,106
- Control Key 18,19,26,40

- Database 1,7,81,106
- Decoding 30,88
- Default 16,24,32,46
- Delete 62,63,64
- Diagnostic Flags 63,100

- Editor 1,7,26,40,84,106
- Editor Error Messages 97
- Empty (null) Entry 22,86,87,91,107
- Equipment and Materials Needed 11

- Filename 106
- Formatting Diskette 11,12

- Getting Started 11
- Glossary 106-110

- Initialize 98
- Input Buffer 86,91
- Input Routine 12
- Insert 13, 66,69,86,87
- Interpreter 1,7,55,69,97,106
- Introduction 1

Key to Symbols 2

Loading Procedures 12

Location 5,42,98

    Description Table 22,82,90

    Descriptions 15,20,24,47,87

    Names 16

    Number 16,46,84

Main Menu 12

Main Menu Glossary 108

Memory Available 13

Message 50,51,65

    Number 46,84

    Text 46,47,65,66,86

    Text Table 81

Movement Pairs 82

Movement Table 21, 42,44,63,82,88,99

Object 5,39,42

    Description Table 39,82

    Descriptions 24,26,39,47,59

    Starting Location Table 26,40,83

    Starting Locations 26,27,40,61,90

Operation 7

Phase I 15

Phase II 39

Phase III 59

Portable Objects (\* of) 76

Preliminary Procedure 8,11

Printer 85

Protection 3

Reference Section 1,81

Save This Adventure 78

Save, Verify, and Load Database 13

Selling Your Adventure 9

Sounds 53,103,104

Source of Light 25,59,83,89

Standard Design Procedures 1,7

Status Table 53,73,84,98

Synonyms 5,28,29,35,63,82

Syntax Error 13,107

## Table

AdventureWriter Message 84

Location Description 22,90

Message Text 81

Movement 21,42,44,63,82,88,99

Object Description 39,82

Object Starting Location 26,40,83

Status 53,73,84,98

VocabularyAction 31,48,66,83,90,99

VocabularyText 81

Test This Adventure 56

Tutorial 1

## Using This Manual 1

Vocabulary 5,28,29,31

Action Table 31,48,66,83,90,99

Text 42,61

Text Table 81

Word Value 30,58,73,81,82,91

## NOTES

## NOTES

## NOTES



